Internet of Things

Souvik Pal
Claudio Savaglio
Roberto Minerva
Flávia C. Delicato *Editors*

IoT Edge Intelligence

Springer

Internet of Things
Technology, Communications and Computing

Series Editors
Giancarlo Fortino
Rende (CS), Italy

Antonio Liotta
School of Computing, Edinburgh Napier University, Edinburgh, UK

The series Internet of Things - Technologies, Communications and Computing publishes new developments and advances in the various areas of the different facets of the Internet of Things. The intent is to cover technology (smart devices, wireless sensors, systems), communications (networks and protocols) and computing (theory, middleware and applications) of the Internet of Things, as embedded in the fields of engineering, computer science, life sciences, as well as the methodologies behind them. The series contains monographs, lecture notes and edited volumes in the Internet of Things research and development area, spanning the areas of wireless sensor networks, autonomic networking, network protocol, agent-based computing, artificial intelligence, self organizing systems, multi-sensor data fusion, smart objects, and hybrid intelligent systems.

Indexing: *Internet of Things* is covered by Scopus and Ei-Compendex
**

Editors
Souvik Pal, Claudio Savaglio, Roberto Minerva and Flávia C. Delicato

IoT Edge Intelligence

Springer

Editors
Souvik Pal
Department of Computer Science and Engineering, Sister Nivedita University (Techno India Group), Kolkata, West Bengal, India

Claudio Savaglio
Department of Computer Science, Modeling, Electronics and Systems Engineering (DIMES), Università della Calabria, Rende, Italy

Roberto Minerva
Institut Mines-Telecom, Telecom SudParis, Evry, France

Flávia C. Delicato
Institute of Computing, Fluminense Federal University, Niteroi, Brazil

ISSN 2199-1073 e-ISSN 2199-1081
Internet of Things
ISBN 978-3-031-58387-2 e-ISBN 978-3-031-58388-9
https://doi.org/10.1007/978-3-031-58388-9

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2024

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

As the world grapples with the challenges posed by the sheer volume of data and the limitations of traditional cloud computing, we shine a spotlight on Edge Computing as a paradigm that not only addresses these challenges but also paves the way for unprecedented innovation. We explore how the synergy of edge computing and IoT brings intelligence closer to the source, enhancing decision-making capabilities and expediting processes, particularly in sectors like automation and industrial IoT.

This book is crafted to cater to a diverse audience, including specialists, researchers, graduate students, engineers, and professionals deeply engaged in research on edge computing, IoT edge, Internet of Things, artificial intelligence, machine learning, and their myriad applications. By presenting a nuanced understanding of AI-enabled IoT edge paradigms, we aspire to contribute meaningfully to the ongoing dialogue surrounding the future of technology and its potential to uplift humanity. This book covers a wide range of AI-enabled edge technologies. This book aims to explore the insight paradigm of the edge technologies which will provide a smooth platform for the scope of industry-academia. The contents include functional framework and protocols for edge computing-based system; intelligent object identification; intelligent sensors; learning and analytics in intelligent IoT-enabled systems; intelligent transportation system; service provisioning at the edge; time series anomaly detection for edge computing applications; modeling, simulating, and evaluating complex end-to-end edge intelligence systems; edge intelligence in autonomous navigation systems; IoT-edge intelligence in healthcare systems; secure data exchange in edge-cloud environment; security, privacy, trust, and provenance issues in IoT-edge environment; and secure neural network inference for edge intelligence. The chapters within this book are meticulously organized to facilitate a seamless exploration of concepts, technologies, and groundbreaking findings. We invite you to embark on this intellectual journey with us as we navigate the frontiers of IoT edge intelligence and envision a future where innovation at the edge transforms the way we interact with the digital universe.

Souvik Pal

Claudio Savaglio

Roberto Minerva
Flávia C. Delicato
Kolkata, West Bengal, India
Rende, Italy
Evry, France
Niteroi, Brazil

About the Book

The purpose of this book is to explore the development of upcoming technologies driven by artificial intelligence within the realms of cloud computing, edge computing, and the Internet of Things (IoT). The primary objective of this publication is to consolidate various technologies into a unified platform, making it easily comprehensible for undergraduate and postgraduate students, researchers, academics, and industry professionals. The focus of this volume is on facilitating a clear understanding of IoT edge intelligence, edge computing, device-edge, and learning analytics in edge-enabled scenarios. The book employs principles of data and network engineering along with intelligent decision support system design to create a dependable IoT edge-cloud ecosystem and implement solutions for cyber-physical pervasive infrastructure.

Numerous organizations and enterprises are embracing compact and mobile devices, along with sensors and actuators that furnish data and services to users or other devices. These devices are poised to become increasingly sophisticated as technology advances. Coping with the substantial volume of data, commonly known as big data, presents challenges in terms of transfer from/to remote and, at times, distant sources. Cloud infrastructure currently stores and processes this data; however, drawbacks such as considerable internet traffic, constrained network bandwidth, elevated latency response, and security concern accompany cloud computing. To address these issues, edge computing emerges, coupled with essential and potential intelligent management and control at the edge. In contrast to the cloud, which serves as an IT environment for abstracting, pooling, and sharing resources across a network, the edge is a computing location composed of hardware beyond the datacenter. At the edge, data are collected and processed, and workloads are offloaded to individual devices on the network’s periphery, as seen in IoT edge. Enterprises have been strategically bringing analytics and business intelligence closer to data generation points in recent times. Edge computing solutions position edge devices, intelligent IoT devices, gateways, and computing infrastructure as close as possible to the data source, facilitating data-driven decision-making. In sectors like automation, intelligent edge devices boast learning analytics and data analytics capabilities, potentially expediting processes in industrial IoT. The forthcoming book will guide readers through the landscape of AI-enabled IoT edge technologies, commencing with an exploration of the insight paradigm and its diverse applications. The book targets researchers and practitioners, offering insights into design architecture, AI algorithms, learning analytics through IoT edge, device-edge, and cutting-edge cloud-IoT countermeasures. The comprehensive discussion covers functional framework and knowledge hierarchy for edge computing; cognitive radio; signal processing; edge access networks and Systems; services and software for IoT edge; platforms and systems for IoT edge; CPU/GPU and hardware for IoT edge; AI-enabled edge computing; blockchain and zero trust in IoT edge; edge-centric convergence of IoT and cloud computing; collaborative edge and cloud computing; storage and caching at the edge; QoS and QoE at the edge; artificial intelligence at the edge; machine learning at the edge; secure, reliable, and trusted edge computing and communications; and networks and communications at edge devices.

The book, structured with a thoughtful chapter organization, aims to convey the concepts of related technologies and novel research findings. By delving into AI-enabled IoT edge paradigms, the book envisions contributing to the betterment of humanity in the future. It serves as a valuable resource for specialists, researchers, graduate students, engineers, and professionals engaged in research on edge computing, IoT edge intelligence, Internet of Things, artificial intelligence, machine learning, and their applications.

Key Features

1. Addresses the complete functional framework and knowledge hierarchy for edge computing and IoT edge.

2. Explores basic and high level concepts, thus serving as a manual for those in the industry while also helping the beginners to understand both basic and advanced aspects in edge-enabled technologies.

3. Based on the latest technologies, and covering the major challenges, issues, and advances in IoT edge environment.

4. Exploring intelligent object identification and object discovery through edge computing ecosystem and its implications to the real world.

5. Explains concepts of edge access networks and systems, services and software for IoT edge, platforms and systems for IoT Edge, CPU/GPU and hardware for IoT edge, AI-enabled edge computing for betterment of the smarter humanity.

6. Exploring intelligent data processing, deep learning frameworks, artificial intelligence at the edge, machine learning at the edge in edge-enabled ecosystem.

7. Exploring edge-centric convergence of IoT and cloud computing, collaborative edge and cloud computing, storage and caching at the edge, QoS and QoE at the edge in edge-based environment.

8. Exploring security and privacy issues and trustworthy security algorithms related to data-intensive technologies in edge-enabled ecosystem.

Contents

Part I Architecture, Systems, and Services

Modeling, Simulating, and Evaluating Complex End-to-End Edge Intelligence Systems
Harikrishna Kuttivelil and Katia Obraczka

Exploring Edge AI Inference in Heterogeneous Environments: Requirements, Challenges, and Solutions
Roberto Morabito and Mung Chiang

Artificial Intelligence-Enabled Edge Computing: Necessity of Next Generation Future Computing System
Anand Kumar Mishra, R. Ravinder Reddy, Amit Kumar Tyagi and Micheal Olaolu Arowolo

Artificial Intelligence-Based IoT-Edge Environment for Industry 5.0
Souvik Pal

Service Provisioning at the Edge: An AI Approach Based on Policies
Zakaria Maamar, Amel Benna, Noura Faci and Dalal Direm

Unsupervised Time Series Anomaly Detection for Edge Computing Applications: A Review
Danlei Li, Nirmal Nair and Kevin I-Kai Wang

Part II Security and Privacy Paradigm

Secure, Trusted, Privacy-Protected Data Exchange in an Edge-Cloud Continuum Environment
Salvador Cuñat, Matilde Julian, Andreu Belsa, Clara I. Valero, Manuel Esteve and Carlos E. Palau

Security, Privacy, Trust, and Provenance Issues in Internet of Things–Based Edge Environment
Anand Kumar Mishra, Shrikant Tiwari, Amit Kumar Tyagi and Micheal Olaolu Arowolo

Secure Neural Network Inference for Edge Intelligence: Implications of Bandwidth and Energy Constraints
Jorit Prins and Zoltán Ádám Mann

Part III Applications

Internet of Things in Intelligent Transportation Systems
Ghalia Nassreddine, Amal El Arid and Mohamad Nassereddine

IoT-Driven Analytics and Edge Intelligence in Autonomous Navigation Systems
Wilson Nwankwo, Joy Orukwo, Chinecherem Umezuruike, Chukwuemeka Pascal Nwankwo, Paschal Uchenna Chinedu and Chukwuemeka Obasi

Edge-AI for Monitoring Air Pollution from Urban Waste Incineration: A Survey
Fatou Ciré Emilie Sané, Maissa Mbaye and Bamba Gueye

Enabling Artificial Intelligence on IoT Edge: Smart Approaches and Solutions for Providing Remote Dental and Medical Services
Aya Sedky Adly, Afnan Sedky Adly, Eve Malthiery, Shahid Ali Shah, Elias Estephan and Mahmoud Sedky Adly

Index

About the Editors

Souvik Pal

is an Associate Professor in the Department of Computer Science and Engineering at Sister Nivedita University (Techno India Group), Kolkata, India. Prior to that, he was associated with Global Institute of Management and Technology; Brainware University, Kolkata; JIS College of Engineering, Nadia; Elitte College of Engineering, Kolkata; and Nalanda Institute of Technology, Bhubaneswar, India. Dr. Pal received his MTech and PhD degrees in the field of Computer Science and Engineering from KIIT University, Bhubaneswar, India. He has more than a decade of academic experience. He is author or co-editor of more than 15 books from reputed publishers, including Elsevier, Springer, CRC Press, and Wiley, and he holds 3 patents. He is serving as a Series Editor for Advances in Learning Analytics for Intelligent Cloud-IoT Systems, published by Scrivener-Wiley Publishing (Scopus-indexed); Internet of Things: Data-Centric Intelligent Computing, Informatics, and Communication, published CRC Press, Taylor & Francis Group, USA; and Conference Proceedings Series on Intelligent Systems, Data Engineering, and Optimization, published CRC Press, Taylor & Francis Group, USA; Dr. Pal has published a number of research papers in Scopus/SCI/SCIE journals and conferences. He is the organizing chair of RICE 2019, Vietnam; RICE 2020, Vietnam; ICICIT 2019, Tunisia. He has been invited as a keynote speaker at ICICCT 2019, Turkey, and ICTIDS 2019 and 2021, Malaysia. He has also served as Proceedings Editor of ICICCT 2019 and 2020; ICMMCS 2020 and 2021; ICWSNUCA 2021, India. His professional activities include roles as Associate Editor, Guest Editor, and Editorial Board Member for more than 100+ international journals and conferences of high repute and impact. His research area includes cloud computing, big data, Internet of Things, wireless sensor network, and data analytics. He is a member of many professional organizations, including MIEEE; MCSI; MCSTA/ACM, USA; MIAENG, Hong Kong; MIRED, USA; MACEEE, New Delhi; MIACSIT, Singapore; and MAASCIT, USA.

Claudio Savaglio

is a Tenure-Track Assistant Professor in the Department of Computer Science, Modeling, Electronics and Systems Engineering (DIMES) of the Università della Calabria, and a Researcher at ICAR-CNR (institute for high-performance computing and networking of the Italian National Research Council), Italy. He received the Master degree in Computer Engineering (2013) and the PhD degree in ICT (2018) both from the University of Calabria (Unical), Italy, where he was also Postdoctoral Research Fellow. In 2020, he obtained the Italian National Scientific Habilitation to Associate Professor in information processing systems. Dr. Savaglio was a Visiting Researcher at the University of Texas at Dallas (TX, USA) in 2013, at the New Jersey Institute of Technology (NJ, USA) in 2016, and at the Universitat Politècnica de València (Spain) in 2017. His research interests include the Internet of Things (IoT), network simulation, edge intelligence, and agent-oriented development methodologies. He has co-authored 70+ scientific papers on the aforementioned topics which have been published in international journals, book chapters, and conference proceedings. He currently serves as Associate Editor for the *IEEE Transactions on Systems, Man and Cybernetics: Systems* andGuest Editor for the *Journal of Computers and Electrical Engineering* (CAEE), *Sensors*, and *IET* journal. He has also served as a technical program committee member and reviewer for many premium conferences and journals in the areas of IoT, computer communications, and networks. He is a member of the PhD board in ICT at University of Calabria and of the Italian PhD board in Autonomous Systems (DAUSY).

Roberto Minerva

(Senior Member, IEEE) received the MS (summa cum laude) degree in Computer Science from the University of Bari, Italy, in 1987, and the PhD degree in Computer Science and Telecommunications from Pierre and Marie Curie University-Sorbonne University, Paris, France, in 2013. From 1987 to 1996, he was a Researcher in the area of service architectures and network intelligence within the Telecom Italia Research Center. In the following years, he was appointed responsible of several research groups related to network intelligence and evolution to next-generation networks. From 2013 to 2016, he was appointed to the Strategic Initiatives of TIM. In

2016, he was the Technical Project Leader of SoftFIRE and a European Project devoted to the experimentation of NFV, SDN, and edge computing. Since 2018, he has been an Associate Professor of Softwarization within the Service Architecture Lab of the Wireless Networks and Multimedia Services Department of Institut Mines Telecom Sud Paris, part of Institute Polytechnique de Paris, Evry, Paris, France. He is the author of the book *Networks and New Services: A Complete Story*. He is the author of more than 50 articles in journals and international conferences. Dr. Minerva has been the Chairperson of the IEEE IoT Initiative in the period (2014–2016). He is currently a member of the Scientific Committee of the Fondazione Bruno Kessler, Trento, Italy.

Flávia C. Delicato

is an Associate Professor at the Fluminense Federal University, Brazil. Her main research interests are edge/fog computing, Internet of Things, adaptive middleware, sensor, and network virtualization. She has 2 books and over 230 published papers. She coordinates and participates in R&D projects funded by companies such as EMC, Dell, and Petrobras. Her research has also received funding from Brazilian and international development agencies, including CNPq, RNP, FAPESP, FAPERJ, CENPES, MCT, Fundacion Carolina, and The Australian Agency for International Development (AusAID). She has served as General and Program Chairs at several national and international conferences. She serves/has served as Associate Editor of the following journals: *Ad hoc Networks*, *ACM Computer Surveys*, *IEEE Transactions on Service Computing*, *IEEE Open Journal of the Communications Society*, *ITU Journal on Future and Evolving Technologies*, and *Journal of Interconnection Networks*, besides acting as Guest Editor for Special Issues of journals such as *Future Generation Computer Systems* and *IEEE Transactions on Consumer Electronics*. Since 2017 she is a member of the IEEE Technical Committee on Smart World—SWTC (http://www.cybermatics.org/swtf/indexTC.html). Since 2020, she has been one of the Chairs of the IEEE Special Committee on Hyper-Intelligence (Hyper-Intelligence Technical Committee—HITC—https://ieee-hyperintelligence.org/member).

OceanofPDF.com

Part I
Architecture, Systems, and Services

OceanofPDF.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
S. Pal et al. (eds.), *IoT Edge Intelligence*, Internet of Things: Technology, Communications and Computing
https://doi.org/10.1007/978-3-031-58388-9_1

Modeling, Simulating, and Evaluating Complex End-to-End Edge Intelligence Systems

Harikrishna Kuttivelil[1] ✉ and Katia Obraczka[1] ✉
(1) University of California, Santa Cruz, CA, USA

✉ **Harikrishna Kuttivelil (Corresponding author)**
Email: hkuttive@ucsc.edu

✉ **Katia Obraczka**
Email: katia@soe.ucsc.edu

Keywords Modeling – Evaluation – Simulation – Edge intelligence systems

1 Introduction

The Internet's edge has been experiencing an ever-increasing proliferation of applications and services running on a wide range of connected devices equipped with a multitude of sensors. The so-called Internet of Everything (IoE) is of increasing technical, social, and economic significance, connecting people, content, processing, and storage resources with the physical and cyber worlds. Projections for the impact of IoE deployments are impressive, with some anticipating as many as 100 billion connected IoE devices and a global economic impact of more than $11 trillion by 2025 [1]. IoE devices at the Internet's edge generate large amounts of data that drive numerous machine learning-powered applications enabled by the cloud. The traditional model for edge intelligence systems is to have client devices transmit their data directly to some central server that runs the intelligence processes. Modeling and simulation of these systems are fairly established. However, as centralized intelligence systems limit scalability and performance and as edge devices become more computationally potent, research and development has been shifting toward intelligence systems that are distributed, decentralized, and closer to the edge of the network following a paradigm shift known as edge intelligence.

Edge Intelligence

Edge intelligence (EI) has been defined as the confluence of edge computing and artificial intelligence [2]. An edge intelligence system (EIS)—a system that supports and utilizes edge intelligence—draws from these fields and from many other fields, depending on the driving of applications. As such, edge intelligence systems also inherit the modeling, simulation, and evaluation challenges from their driving applications. The development and proliferation of EIS have resulted in a plethora of new applications and solutions to challenging problems—including, but certainly not limited to, smart home systems, autonomous vehicle fleets, intelligent manufacturing systems, natural disaster detection systems, and smart agriculture [2–4]. Such deployments range from having homogenous devices connected over homogeneous networks executing simple machine learning (ML) algorithms to heterogeneous, dynamic networks of heterogeneous devices with a range of capabilities executing complex or optimized ML algorithms. Whether simple or complex, the design, development, and modeling of edge intelligence systems requires the integration of different physical and functional components.

Compared to more traditional intelligence systems, the inherent complexity of edge intelligence systems necessitates a deeper understanding of their components and structures. Designing, modeling, and implementation is an iterative process in which we start with designing our system to try and meet performance requirements, create representative models of the edge intelligence system, implement it through tools capable of observing our desired metrics, and then evaluate our implemented model to check whether or not it meets our performance requirements.

Modeling, Simulation, and Evaluation

This chapter discusses the different aspects and issues involved in modeling, simulation, and evaluation of edge intelligence systems. Modeling involves building a representation and its underlying components—often simplified with only relevant features of interest—and the interactions that occur between those components. Simulation involves the execution of a system model using physical or software platforms that allow observations of the system model under different scenarios or conditions. Evaluation allows us to assess and gain insight from those observations using carefully selected metrics that help create a perspective of the system as a whole. Typically, modeling is seen as the basis of the process because it results in a model, which is then executed through simulation, and then observed in evaluation. However, there is also a lot of backpropagations present when designing solutions for modeling, simulating, and evaluating edge intelligence systems. Metrics planned for evaluation must be considered when selecting tools and programs for simulation and the level of depth or complexity of our modeling. Tools, platforms, and programs for simulation can often enable or limit the depth, complexity, and completeness of our modeling and the selection of metrics for evaluation. Ultimately, how we model our system itself is dependent on how we plan to execute the model through simulation and what metrics we are interested in for evaluation. Given this interconnectedness, we proceed with our discussions in this chapter with a holistic view of designing solutions for testing edge intelligence systems. We will discuss these notions as part of the larger processes they contribute to.

Chapter Roadmap

The remainder of the chapter is organized as follows: In Sect. 2, we define and categorize the components of edge intelligence systems. Sect. 3 discusses the different performance metrics relevant to the modeling, simulation, and evaluation of edge intelligence systems. In Sect. 4, we explore the landscape of tools, libraries, and environments that enable the modeling, simulation, and evaluation of the different components and aspects of edge intelligence systems. Finally, in Sect. 5, we expand our exploration into integrated, holistic solutions for the simulation of edge intelligence systems.

2 Components of Edge Intelligence Systems

We describe the components of an edge intelligence system illustrated by a climate-resilient agriculture application scenario, more specifically a simplified version of intelligent greenhouse food production [5]. Consider that we are researchers in a multi-institution system that intends to design and develop a smart greenhouse system in which several greenhouses are connected via the Internet to each other and to a center of operations to enable collaborative intelligence for predictive and analytic services. Within each smart greenhouse, a wireless router connects several microcontrollers, each equipped with configurable sensors to collect temperature, humidity, and soil moisture data and actuators to control irrigation and light. These sensor-enabled, wireless-enabled microcontrollers—sensor tags—are connected through a wireless router to a server within the greenhouse. Operators can connect to the server through a workstation with a wired connection to the greenhouse server. In the center of operations, a secure workstation is connected to the central server with a wired connection. The intention of this system is to have each greenhouse server train or update machine learning models on data collected through the sensor tags to predict future conditions for the purpose of analytics and to adjust sensor polling, irrigation, or light levels responsively. The system as a whole employs federated learning [6] such that: (1) the central server disseminates the machine learning models to the greenhouse servers; (2) the greenhouse servers train these machine learning models on their collected data and update them; (3) the greenhouse sends the model updates to the central server; and (4) the central server aggregates these model updates to update its own machine learning models and disseminates the new, updated machine learning models to the greenhouse networks. A high-level overview of this edge intelligence system is shown in Fig. 1.

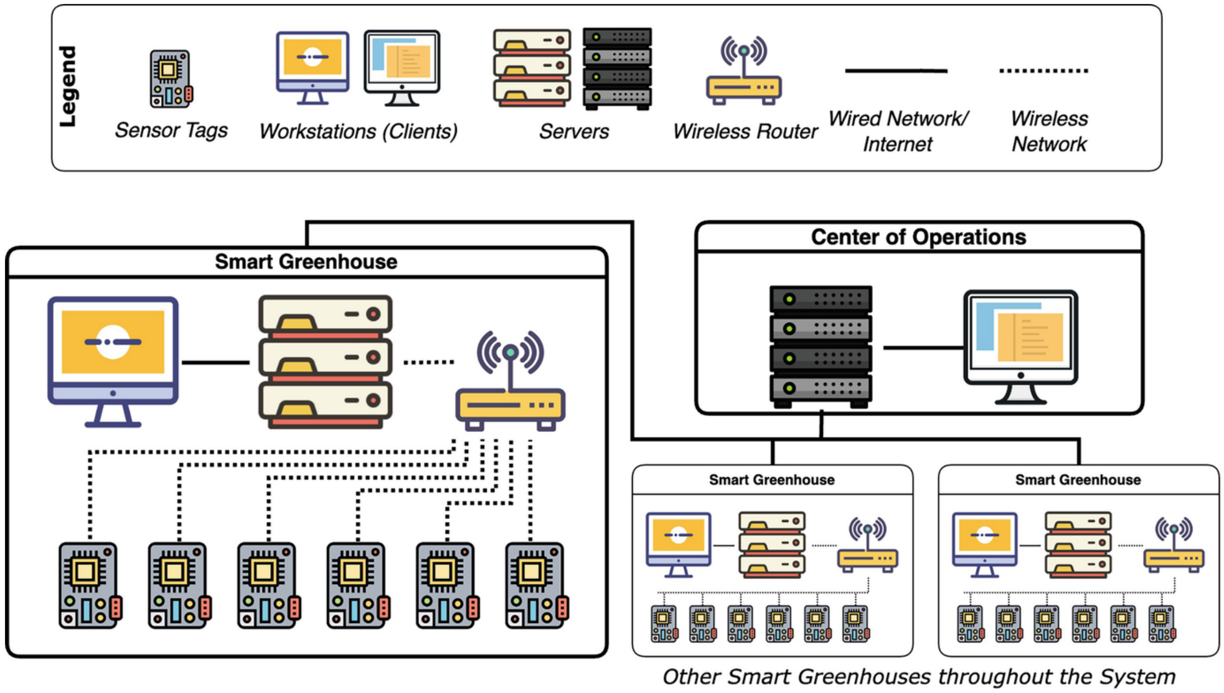

Legend
Sensor Tags
Workstations (Clients)
Servers
Wireless Router
Wired Network/ Internet
Wireless Network
Smart Greenhouse
Center of Operations
Smart Greenhouse
Smart Greenhouse
Other Smart Greenhouses throughout the System

Fig. 1 High-level overview of a smart greenhouse system, a simple example of an edge intelligence system

When edge intelligence systems are designed and modeled, various considerations must be taken into account. Using the smart greenhouse system as our working example, the following questions need to be considered: How will data—and what kind of data will—be exchanged between the sensor tags and the greenhouse server, the greenhouse servers and the central server, and the workstations and the servers? How do operators and applications access raw data, processed data, ML models, and ML output? What ML models can train and infer effectively from the data sourced from the sensors and are able to meet the requirements of the driving application(s)? Which devices and platforms are capable of which computational or communication processes? Questions would also arise about the operations that need to take place within the system as a whole and the local greenhouses. Where would data be stored or replicated for reliable and fast access? Where would it be stored for both inference and training purposes? How often do we need to carry out inference and training processes? Where are the ML models executed, and how are model updates exchanged?

These questions usually come up in the design phase of the edge intelligence system but carry over into the modeling phase, as the system model should be a simplified—yet realistic—representation of the real system. While the implementation of edge intelligence systems—from distributed greenhouses to automated vehicle fleets to industrial Internet-of-Things—can vary greatly, there are common components and design goals that apply to all. We partition edge intelligence system components into two dichotomies—*infrastructure components* and *functional components*. We use the term “infrastructure components” to describe the physical components—i.e., the network devices and transmission media that connect them—and the end devices or systems. We use the term “functional components” to refer to the logical and application components of the system, namely communication protocols, application functions (e.g., machine learning, data processing, orchestration), and information exchanges that support the intelligence functionality and are executed on the infrastructure components. In the remainder of this section, we explore how these foundational building blocks can be used to model the entire system. Later in this chapter, we will explore the tools, platforms, and services that allow us to implement, simulate, and evaluate edge intelligence systems and their components.

2.1 Infrastructure Components

When considering the infrastructure of an edge intelligence system, the main components include:

- The network devices and transmission media that connect the end devices and systems and facilitate communications between them; and
- The end devices and systems that are connected through the edge network are the platforms in which the functional components are deployed.

Figure 2 shows the infrastructure components of our example smart greenhouse system, whose considerations we will follow as we discuss the modeling of infrastructure components (Fig. 3).

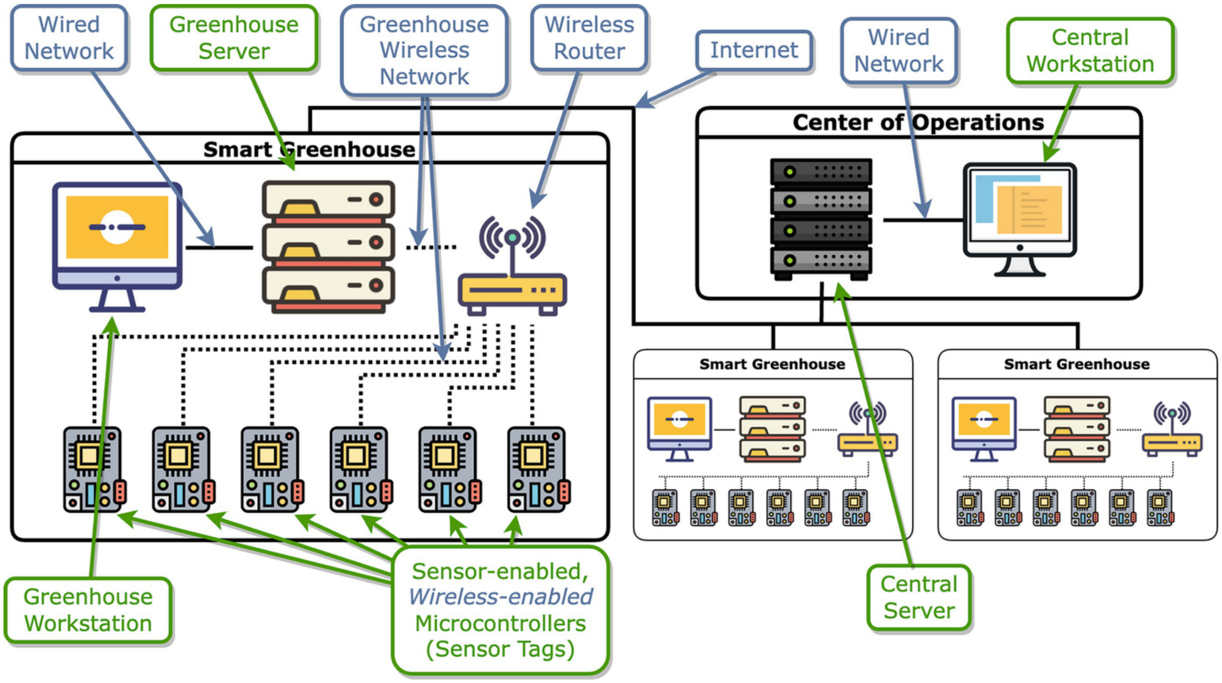

Wired Network
Greenhouse Server
Greenhouse Wireless Network
Wireless Router
Internet
Wired Network
Central Workstation
Smart Greenhouse
Center of Operations
Smart Greenhouse
Smart Greenhouse
Greenhouse Workstation
Sensor-enabled, *Wireless-enabled* Microcontrollers (Sensor Tags)
Central Server

Fig. 2 Infrastructure components of the smart greenhouse system, with network components labeled in blue and end devices and systems labeled in green

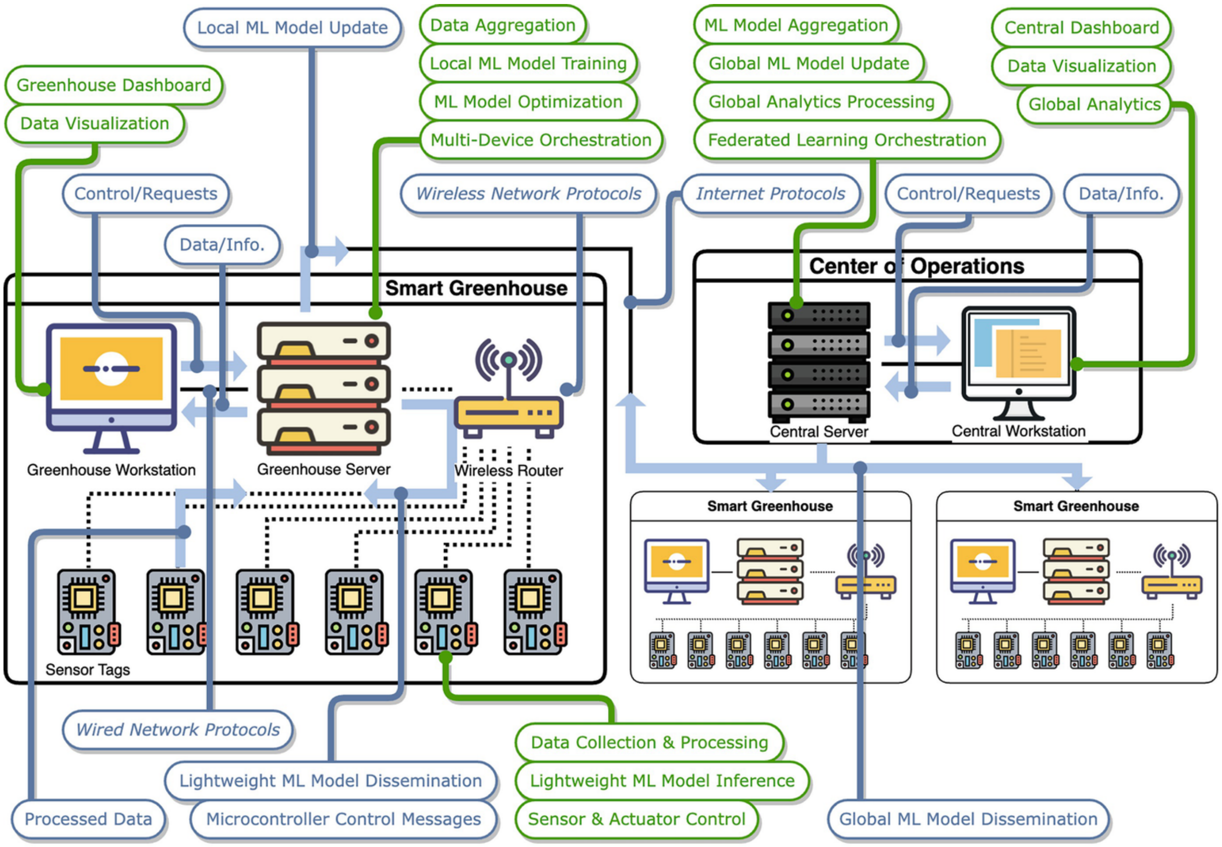

Local ML Model Update
Data Aggregation
Local ML Model Training
ML Model Optimization
Multi-Device Orchestration
ML Model Aggregation
Global ML Model Update
Global Analytics Processing
Federated Learning Orchestration
Central Dashboard
Data Visualization
Global Analytics
Greenhouse Dashboard
Data Visualization
Control/Requests
Data/Info.
Wireless Network Protocols
Internet Protocols
Control/Requests
Data/Info.
Smart Greenhouse
Center of Operations
Greenhouse Workstation
Greenhouse Server
Wireless Router
Central Server
Central Workstation
Smart Greenhouse
Smart Greenhouse
Sensor Tags
Wired Network Protocols
Data Collection & Processing
Lightweight ML Model Dissemination
Lightweight ML Model Inference
Processed Data
Microcontroller Control Messages
Sensor & Actuator Control
Global ML Model Dissemination

Fig. 3 Functional components of the smart greenhouse system, with communication protocols and data exchanges labeled in blue and intelligence and data processes labeled in green

The Network When modeling a network, it is useful to consider the basic network stack: the application layer, the transport layer, the network layer, the link layer, and the physical layer. However, in the context of edge intelligence systems specifically, and given that some common network technologies are implemented in a way that spans and abstracts some of these layers, we can reduce our scope to the essentials of our functional network. One should consider the physical media that physically form the network, e.g., whether the network is wired, optical, or wireless. One should also consider the physical environment that can affect the connectivity of the physical media, e.g., the effect of trees obstructing the line-of-sight between wireless antennas or mobility patterns that change the connectivity between two points over time.

Using our example of the smart greenhouse system, the researchers need to consider how the sensor tags will connect to the greenhouse server over the wireless network and how greenhouse servers in different areas connect to each other and the center of operations. Issues such as the availability of resources, the cost of additional infrastructure, public and organizational policies, and more affect these decisions. In our example, we will presume that the sensor tags within the greenhouse would use a wireless network in a relatively unobstructed environment to connect with a greenhouse server via a wireless router. Each greenhouse server would be connected via wired Ethernet connections to the Internet in order to communicate with the central server. The secure workstations in both the greenhouses and the center of operation would be able to access the servers through wired Ethernet connection.

End Devices and Systems End devices are devices that are the terminal nodes of a network. These devices are typically the sources and destinations of data and range from sensor nodes to servers, clouds, laptops, storage devices, and workstations. We use “end systems” to refer to abstractions of external systems or subsystems connected to the edge intelligence system. In edge intelligence systems, these end devices and systems can range from sensors or sensor-enabled devices to cloud-based offloading services.

When modeling end devices, the main considerations are computational capability, connectivity, power consumption, and reliability. In many cases, determining how to model the computational capabilities of end devices can take a high-level perspective, e.g., whether or not a specific end device can run specific functions, protocols, or operations. In the cases where advanced modeling is required—e.g., incorporating the use of hardware emulators—we must determine the hardware specifications, such as the processor type (e.g., CPU, GPU, or the recently termed Neural Processing Unit, or NPU, for ML operations) and specifications (number and types of cores, clock speed, etc.), memory and memory hardware, and storage and storage hardware. Modeling connectivity requires determining when devices connect with other devices over the network and how reliable those connections are. Identifying real world patterns or functions of behavior—e.g., mobile devices entering and leaving the transmission ranges of a cellular network or predictable network congestion due to the observation of regular patterns—plays an important role in modeling device connectivity. Modeling power consumption in its simplest form is using a linear function of time, such that while a device is presumed to be on and functioning, it consumes power at a consistent rate and depletes it from a set capacity battery source if the device is battery-powered. A more advanced take on modeling power consumption might require a nonlinear function of not just time but also the computational workload of operations taking place on the device, conditions in the environment (e.g., operating temperatures), charging and discharging rates for mobile devices and associated power sources and batteries, battery and power source specifications, and more. At the highest level is the reliability of the device to the edge intelligence system, e.g., considering whether or not the device would be functioning predictably and expectedly, whether it will be affected by detrimental conditions in the environment and to what extent, or considering whether the device can remove itself entirely from the network, among other considerations. End devices can vary greatly in their function, and each function may require different types of models. With sensors and input devices, it is important to determine the type and size of data that devices collect, the polling and transmission behavior of that, and the connectivity to the rest of the system, e.g., a remote temperature sensor array that collects temperature data as an array of double float precision numbers to represent the value in Celsius, is 200 bytes per reading, polls at an adjustable rate with a default rate of 1 kHz, transmits the aggregated values at a rate locked to the polling rate, and connects to a microcontroller device with connections for power, data, and control, allowing for the microcontroller to adjust the polling and transmission rates of the sensor array.

When modeling end systems, we must consider many of the same considerations for end devices, albeit from a more high-level, abstracted perspective. This chapter opts to distinguish between end *devices* as devices of which we have ownership and relevant knowledge of their internals and end *systems* as external systems for which we do not have such ownership and therefore must rely on external specifications or “black box” thinking, i.e., the inputs and outputs of such systems. In considering computational capabilities, the specifications—such as workload processing speed, available memory, and available storage—provided by the producer of the end system or third-party experiments and reviews done on the end system can provide useful information on the expected behavior to model. In considering connectivity, one must understand the connection constraints—e.g., a cloud-based storage service capping upload and download speeds or limiting the total upload or download volume for a period of time—and system connection availability—i.e., when the external system is made accessible via the network—which can also help determine the reliability to the rest of the system. Considering power consumption for an external system can be a more complicated task as systems are complex, and therefore a black box approach may not yield sufficient similarity to real-world behavior. However, specifications from first-party or third-party evaluations are sometimes available and can provide the relevant information needed to model power consumption.

In our smart greenhouse system example, our end devices include the sensor tags, the servers, and the workstations. For the sake of this example, we will presume that there is sufficient attached storage for each of these devices. For the sensor tags, we would have to consider the functions and connections of the sensors attached to the microcontroller, the computational capability of the microcontroller itself, the power consumption of the microcontroller (and whether it is powered by a plugged-in power source or by a battery), and the connectivity to the greenhouse server. We can presume that the sensor arrays collect data in 1 KB packets at a rate of 2 Hz. Given the network infrastructure, we can assume that connectivity is not a significant issue, but that this data is able to be partially processed and aggregated at the microcontroller itself and sent over the wireless network at a rate of 1 Hz to the server. We can presume that the sensor tags use batteries that have limited capacity before needing to be replaced, while the workstations and servers are connected to power sources. Similar considerations must be given to the servers and workstations. We can presume that the servers themselves are able to process and send information and run operator-facing dashboards without any significant constraints. With considerations like this and those we made with the network infrastructure, we have a sufficiently complete model of the infrastructure of the edge intelligence system.

2.2 Functional Components

Identifying the functional components of an edge intelligence system is guided by considering the system's overall functional purpose—providing intelligence at the edge. In most intelligence systems, the data is of critical importance, and thus data collection and processing are essential functional components of edge intelligence systems. Another important consideration is which intelligence model(s) best suit the system's target application(s) and collected data and what framework and tools can be used to implement the intelligence model(s) within the constraints while satisfying the requirements of the system. Finally, we must consider how end devices and systems communicate to enable, build, and share intelligence. As such, the main functional components of an edge intelligence system include:

- *Data collecting and processing*, ranging from collection to aggregation to preprocessing for model training;
- *Intelligence that drives the system objectives*, the artificial intelligence and machine learning models and algorithms that use the data to train and service the needs of the driving application; and
- *Communication protocols and information exchange*, the communication protocols that enable information exchange and the information exchanges that enable end-to-end communication and edge intelligence throughout the system.

Modeling the Scenario Through Data Intelligence starts with data, either collected and pre-trained a priori to system deployment or actively collected and trained online during the runtime of the system. In the various driving applications of edge intelligence systems, different scenarios involve interfacing with different types of data. When we consider the environment that an edge intelligence system is deployed in, we think of the physical infrastructure and environmental factors that affect that infrastructure. Similarly, as we consider the scenario in which the edge intelligence system is deployed, we consider what the system experiences as input. Therefore, while modeling algorithms and physical systems involves modeling hardware and their capabilities and behaviors, how we model scenarios is ultimately how we model the datasets that feed into the edge intelligence system. Hoffpauir et al. [7] identify three key challenge areas with procuring data for edge intelligence systems—data collection, data quality, and data efficiency—that apply to how we model target scenarios.

- *Data Collection*: The challenge with data collection is collecting a large amount of high-quality and efficient data sufficient for use with the machine learning models. This data—especially when collected and aggregated from multiple sources—must be relevant and address the inherent biases, noise, and errors in the data.
- *Data Quality*: The challenge with data quality is collecting accurate, reliable, and complete—i.e., high quality—data. Machine learning models can be sensitive and vulnerable to the lack of quality in the data they train on. Low-quality data contains errors, inconsistencies, and missing data points that can greatly reduce the speed or performance of model training and inference. Where collecting or procuring low-quality data cannot be avoided, additional preprocessing measures must be taken to ensure that the data is cleaned and usable for the machine learning models.
- *Data Inefficiencies*: The challenge with data efficiency is collecting relevant and useful enough data, i.e., data that is efficient for machine learning models to train with. Inefficiencies in the data arise when the data is incomplete in terms of missing or irrelevant data points or features, i.e., data that does not contribute to or negatively affect the machine learning model's training performance. Ideally, irrelevant or useless data should be removed through trimming datasets and feature selection.

To address these challenge areas, most edge intelligence systems loosely follow the process shown in Fig. 4 and described below.

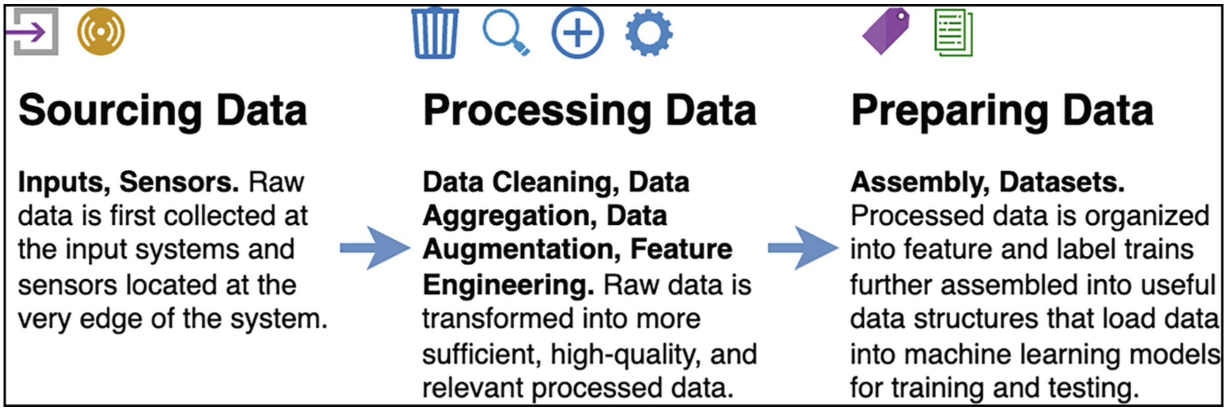

Sourcing Data
Inputs, Sensors. Raw data is first collected at the input systems and sensors located at the very edge of the system.
Processing Data
Data Cleaning, Data Aggregation, Data Augmentation, Feature Engineering. Raw data is transformed into more sufficient, high-quality, and relevant processed data.
Preparing Data
Assembly, Datasets. Processed data is organized into feature and label trains further assembled into useful data structures that load data into machine learning models for training and testing.

Fig. 4 The general process of sourcing, processing, and preparing data for machine learning model training

The process begins with either sensors or inputs from other devices or systems—e.g., physical, digital, or human inputs—sourcing data. This data is then either collected or sampled at either the source itself or transmitted via interface to edge devices. However, raw data is usually not useful for machine learning models as it highly depends on the quality of the data sources; the raw data may contain errors, inconsistencies, missing data points, and other flaws. Therefore, it is necessary to preprocess the raw data before using it for machine learning purposes. In the case of multiple data sources transmitting data to a collection point, preprocessing sometimes includes data aggregation at this collection point through various methods, including reducing the dimensionality across features through feature selection or feature extraction, creating representative bins of data (e.g., averaging sensor values recorded over a period of time into a single value), or acting as a gateway through which data is selectively passed on based on some predefined criteria [8]. Preprocessing also includes addressing the aforementioned quality issues by omitting outlier data points that could be attributed to faulty inputs or sources, reconstructing or omitting missing sequences of data points, or removing noise and other inconsistencies. The final stage of preprocessing transforms the data so that it can interface with the target machine learning models. For example, this includes normalizing quantitative data values for input layers, encoding and embedding text data for natural language processing, or building sequences for use with sequence prediction. The ultimate goal of this last stage is to produce immediately usable feature and label sets for training, validating, testing, and inferring with the machine learning model.

While generally following these steps can provide coverage for addressing challenges in collection, quality, and inefficiency, taking extra care within each of these steps can further mitigate or reduce the loss of performance incurred by these challenges. We propose some guidelines for this extra care below:

1. *Generate or find large, relevant, and high-quality—and, where applicable, distributed and source-attributed—datasets when possible.* The distribution of data among clients and the production of that data from different sources more accurately models the decentralization of data that is present in many, but not all, edge intelligence applications. When it is not possible to source enough high-quality and efficient data, then special methods such as artificial data augmentation [9, 10] or data-efficient machine learning [11] are options.

2. *Determine the processing necessary to make the raw data useful, relevant, and compatible with the driving application and machine learning models.* This includes understanding to what degree the sourced data has already been processed, in addition to identifying the processing required to make that data useful.

3. *Ensure that the data is sufficiently processed and organized to be input into the machine learning models.* This could include the use of methods like categorization encoding, one-hot encoding, normalization, and other transformations.

In our example with the smart greenhouse system, these could include sensor data for temperature, humidity, and soil moisture sensors; digital input data received from weather services; and user input data collected from the interfaces at the control stations. For this particular example, let us narrow the scope of our driving application to temperature prediction using sensor data from the temperature and humidity sensors.

Designing and Implementing Intelligence The *intelligence* aspect of *edge intelligence* comes from the machine learning (ML) processes that use data collected from sensors and inputs to the system. While the development and implementation of ML models are outside the scope of this chapter, the selection of ML models and accompanying operations (e.g., loss functions, optimizers, and encoders) is important to consider as context for modeling intelligence and is largely dependent on the driving application and the available data. Some examples include using recurrent neural networks (RNNs) for sequential and temporal data such as language and signal analysis and convolutional neural networks (CNNs) for spatial data such as image processing and recognition. These and other forms of neural networks, the most pervasive form of machine learning and the basis of deep learning, are based on models containing a high volume of parameters that must be tuned over time through the process of training on collected data before they can be used to infer new data. As such, the modeling of these neural networks from a system level requires considering the size and complexity of the selected ML models, the training or pre-training of these ML models given the availability of data, the evaluation of these ML models during preparation and runtime, the adaptability of these ML models through continuous iteration and improvement, and the logistics of deploying these ML models within the edge intelligence system.

In our smart greenhouse system example, the intention was to have ML models train on and infer collected data within the greenhouses and employ federated learning [6] to collaboratively train these ML models without compromising data privacy, using the central server as the federated learning server and the greenhouse servers as the clients. Since we have already presumed that the greenhouse and central servers are computationally capable, we can presume that the greenhouse servers are capable of training the ML models on the data collected from the sensors and that the central server is capable of running the aggregation of model updates to facilitate federated learning. Since we have already presumed that the microcontrollers in the sensor tags are not as computationally capable or memory capable, we assume that they may not be able to run the full ML models required for edge intelligence system objectives. We can presume that these microcontrollers can instead run optimized, smaller versions of the ML model—with a smaller size and less complexity—to run inference on collected data on-device to make decisions about adjusting the sensor polling and transmission rates or actuating irrigation or lighting. This would require that the greenhouse servers optimize their ML models before disseminating them to the sensor tags.

Communication Protocols and Information Exchange The functional basis of communication in any networked system is the communication protocols used to communicate across network devices and media. As such, one should consider the link, network, and transport protocols that operate on the physical media and facilitate the communication, e.g., OSPF, IP, UDP, and TCP. Many of these protocols are rooted in strictly defined standards that describe the sometimes-complex mechanisms to facilitate communication. Fortunately, many implementations of network technologies—and their representations modeled in network simulators and emulators—are often packaged. For example, the TCP/IP protocol is often implemented atop the Wi-Fi 802.11 standard over wireless media or atop the Ethernet 802.3 standard over wired media. LoRaWAN is a communication protocol that defines both the long-range wireless radio media and the wide area networking protocols that take place on that media [12]. These stacks, especially the more common ones, are usually implemented in their entirety in network simulators, such as in *OMNET++* [13] with *Inet* [14] or in *ns-3* [15], alleviating the researcher or developer to implement them themselves. We introduce and discuss these tools in Sect. 4.

Atop the communication protocols necessary for information exchange between end devices and systems, we must consider what data is being exchanged, how much data is being exchanged, and how often these transmissions occur. The complexity of edge intelligence systems usually means that several different processes are running in parallel across the system with different types of communication between the different devices, including data sharing, ML model information sharing, orchestration, and control messaging, among others. While security and privacy play an essential role in the deployment of edge intelligence systems, we consider these as functions outside system intelligence for the sake of simplicity. We should also note that modern network development, in contrast to older networking practices, has taken to cross-layer protocol development [16], leveraging all layers of the network stack to provide additional benefit to the application layer. While cross-layer protocols in edge intelligence have not yet been widely studied today, it is possible that they may be a standard tomorrow; as such, we should consider both communication protocols and information exchanges when discussing the communication functionality of the systems.

In our smart greenhouse system example, we have control and orchestration messages, the direct sharing of data and aggregated data, the exchange of information relevant to the ML models, and the operator applications like dashboards and analytics. Control messages would be sent from the greenhouse workstations to the greenhouse servers to configure or adjust the sensor tags within the greenhouses, and an acknowledgement or error message would be sent back from the greenhouse server as a reply. Greenhouse servers would pass on these control requests as orchestration commands to the sensor tags, but the servers would also send optimized machine learning models down to the sensor tags. The sensor tags themselves would be sending acknowledgments or other replies to these messages, but they would also be sending raw, aggregated, or processed data that they collected up to the greenhouse servers. In the center of operations, central workstations would send control messages and receive acknowledgement, data, and error messages. The central server would be receiving machine learning model updates and would be disseminating updated global machine learning models. By determining what kind of data is being sent and received along the various end-to-end connections, we have rounded out our model of our example smart greenhouse system.

---

3 Performance Metrics

The performance metrics of any complex system are in some significant part derived from the performance metrics of the components that make up the system, and in the other parts derived from the interoperation of these different components. This is not any different for edge intelligence systems—as a system of various infrastructure and functional components, the performance metrics of an edge intelligence system are based on the network and device infrastructures, the communication and intelligence services, and all those components within its subsystems.

3.1 Component Metrics

We can start to understand the bigger picture of how we observe the performance of edge intelligence systems by focusing on their main components: the network infrastructure, the end devices’ computational capabilities, the communication protocols and information exchanges that happen on the network infrastructure, and the machine learning and artificial intelligence operations that happen on the devices. Let us consider the communication stack—the network infrastructure and the communication protocols and functions that run atop the infrastructure:

- *Network performance.* The key metrics to network performance are end-to-end latency, throughput, and transmission success rate (i.e., the complement to message drop rate), where end-to-end latency and throughput are often combined into the single metric of bandwidth, which describes how much and how fast information can travel over the network [17]. In wired networks, these metrics are affected by the quality of the cables and hardware ports, while in wireless networks, signal-to-noise ratio (SNR) and signal penetration of the various media that wireless signals pass through are key external factors for affecting performance and reliability—e.g., the measure of transmission success. In any network, the network traffic also affects these metrics, as higher utilization of the network can lead to more congestion and delays, therefore leading to higher latency. At a higher level, network coverage (i.e., how much area a network services) and network availability (i.e., how available the network is when in range) are other important metrics to consider.
- *Communication efficiency.* While the network infrastructure incurs limitations in bandwidth availability and reliability, the overall communication performance depends on how we manage those limitations. Communication protocols employed in edge intelligence systems must balance minimalism and efficiency with completeness and reliability. Communication functions—e.g. information sharing, data forwarding, ML model sharing, etc.—must be careful not to over-utilize the network while still maintaining the minimum network utilization to guarantee proper functioning of the system and robustness against disturbances. Ultimately, the main metric to monitor with either communication protocols or functions is the communication overhead these components generate, i.e., the volume of data being transmitted over the network.

These considerations allow us to converge on some general communication metrics to track: *network latency*, *network throughput*, *network bandwidth*, *network reliability*, *network coverage*, *network availability*, and *communication overhead*.

Next, let us consider the logical stack—the device infrastructure and the intelligence operations that take place on the infrastructure.

- *Device Infrastructure.* The key metrics when considering device infrastructure include metrics associated with hardware performance, such as processor architecture, clock speed, cache, memory, and storage; these metrics are often combined into processor execution speed, memory bandwidth, and storage. As network performance can falter with network congestion, device performance can be negatively affected depending on the workload and how it runs on the process architecture. When considering the performance of device infrastructure from a high level as it pertains to edge intelligence systems, we can consider the metrics to be the execution time of target operations, the memory utilization of target operations, and the total storage that these operations require.
- *Data and Intelligence Operations.* As the network infrastructure places limitations on the communication capabilities of an edge intelligence system, the device infrastructure places hardware performance limitations that must be accounted for when measuring the performance of data and intelligence operations [2–4]. The storage available on device infrastructure makes it important to know the size of stored data necessary to train and infer with ML models. The intelligence operations are also limited to what can be done given the processor's performance and available memory. Beyond that, we must also consider the metrics within the intelligence operations, namely machine learning. Metrics in machine learning include the loss defined by functions that evaluate how well machine learning models perform, the accuracy defined by how accurate the ML model inferences are given the bounds of tolerance, and the confidence defined by either the upper and lower bounds of continuous predictions from regression models or the probability of deterministic choices from classification models.

These considerations allow us to converge on some logical metrics to track: *ML model runtime*, *operation runtime*, *size of stored data*, and *memory footprint*.

3.2 Systemwide Considerations and Trade-Offs

Let us revisit our smart greenhouse system example. At first glance, it can seem pretty straightforward to apply what we derive from considering the various performance elements of the different components to determine the target metrics of our system. After all, in an ideal situation, we want all of our wired and wireless network infrastructure to be 100% reliable and to provide the lowest latency and the highest throughput to achieve the best bandwidth possible and we would want our communication protocols and functions to be highly efficient and streamlined. We would want our microcontrollers and servers to have all the power and computing resources available so that we can run all our operations at the highest speed possible while delivering the best possible results in prediction, analytics, and control. However, in real-world edge intelligence systems, constraints on hardware technology, cost, and operating environments mean that we have to consider what is possible, what is effective given the application, and what target metrics we want to value over the others. In other words, we have to consider the trade-offs of optimizing certain target metrics over others. While we have presented a few of these considerations below, each edge intelligence system is different and therefore may present different and more complex trade-offs [3, 18–20].

- *Communication Efficiency vs. Better Machine Learning Performance*. Whether it is collaborative decentralized edge machine learning or simple data forwarding to centralized machine learning, more consistent and timely communication of model information or data leads to better machine learning performance due to more delivered data or more updated model information [6, 21]. However, ensuring more consistent and timely communication may place more burden on network utilization, especially when accounting for communication strategies to overcome the network volatility present in edge networks. We might consider reducing the frequency of communication but that would lead to worse machine learning performance, though this is a trade-off that can be considered if the lower performance still meets application or system requirements.
- *Lower Machine Learning Latency vs. Better Machine Learning Performance*. The speed at which machine learning inferences can vary depending on the machine learning model size and complexity. It is generally known that models with higher size and complexity—i.e., more parameters and flexibility when interacting with data—tend to yield

higher performance in the form of better accuracy and lower error. However, the higher size and complexity of such models incur higher computational overhead, which translates into slower execution times. To address this problem, many strategies and tools have been proposed to reduce the size and complexity of models, usually at the cost of having at least slightly worse accuracies and greater errors [22]. However, depending on the application and system requirements, some degree of lower accuracy and greater error can be tolerated if it is a result of saving valuable time or computing resources.

- *Power Consumption vs. Better Machine Learning Performance*. As machine learning models with higher model size and complexity incur higher computational overhead, they can also incur higher power consumption to achieve more work in less time to reduce latency or increase capability. For systems that are connected to a reliable and cost-effective power source, this may not be an issue. However, for many edge intelligence systems—especially ones that aim to be energy efficient in power-constrained environments—reducing power consumption for some or all devices is important. So, while higher power consumption can be the consequence of higher computational performance, we might need to consider using more power-efficient strategies that may result in lower quality output or a higher latency. These consequences of lower power consumption may or may not be tolerated, depending on the application and system requirements.
- *Lower Network Latency vs. Better Computational Performance*. Computation offloading is a common method of edge computing in which computation necessary for one device is offloaded to a more capable device [23]. In many cases, devices will offload computation to a more capable fog or cloud computing system at the expense of utilizing the network to send tasks and relevant data to the fog or cloud and receiving outputs that are of a higher quality than if they were done locally. However, network constraints can sometimes make consistent computation offloading difficult and therefore make it necessary to determine when offloading is necessary for higher quality outputs, what devices to offload computation to based on the network and device computational capability, and whether computation on-device can produce outputs that are adequate enough for application and system requirements.

- *More Affordable Hardware vs. Better Computational Performance*. Devices all along the spectrum of cost, from cheap microcontrollers to powerful servers, have seen leaps in computational ability. Machine learning has typically utilized the capabilities of the higher end of this spectrum, running large and complex models to achieve high-quality outputs. However, in many real-world edge intelligence systems, cost can be a prohibitive factor in designing and developing solutions. Depending on the application and system requirements, we might opt for using cheaper hardware and employing machine learning and other methods with lower computational overhead that still meet the requirement, or we may determine that higher-quality output is necessary and worth incurring higher costs for the hardware.

How can we consider these trade-offs in target metrics for the smart greenhouse system? If we presume that the wireless networks connecting the sensor tags to the greenhouse servers have low bandwidth, we may have to consider limiting the frequency at which sensor tags upload data to the greenhouse servers while still ensuring that the uploads happen frequently enough to provide enough timely data to keep the machine learning models trained adequately. If we presume that we have a strict budget for purchasing equipment, we may have to consider using cheaper and more efficient sensor tags, servers, and workstations while still ensuring that the end user experience across all the different functions still meets minimum specifications. If we presume that the sensor tags run on low-capacity batteries that must be manually replaced every few months, then we must consider how we can either scale down or optimize hardware or software components to reduce power consumption while still meeting minimum specifications.

In the broad landscape of edge intelligence systems, these may not be the only trade-offs to consider. For example, in autonomous and self-driving vehicle systems, the safety of users and those around them must be considered, e.g., the trade-off between vehicle user comfort and the safety of pedestrians outside the vehicle. Consider another example in decentralized intelligence using personal mobile devices, in which the privacy and autonomy of users must be considered, e.g., the trade-off between developing powerful machine learning models using unrestricted data and protecting the privacy of data while ensuring users have control over access to their data.

4 Simulation and Evaluation Tools

As mentioned in Sect. 2.1, the basic infrastructure components of edge intelligence systems consist of the network infrastructure, the intelligence services, and the devices and interfaces. Modeling, simulating, and evaluating the system requires researchers and developers to have the utilities to model, simulate, and evaluate each of these system components. Section 2 introduces approaches to model these different components of the edge intelligence system.

4.1 Using ML/AI Libraries and Packages

In Sect. 2.2, we broadly introduced some of the machine learning tools used in edge intelligence systems. Given the constraints of running these tools on edge systems, namely volatile communication channels and limited computational resources, it is important to select a framework and machine learning strategy that is either already resource-aware and efficient in working within your system or one that can be adapted to be as such. With the rise in prominence of artificial intelligence and machine learning—especially in deep learning and neural networks—several machine learning frameworks have emerged as capable and effective tools of implementation, with some of those tools being specialized for the emergence of edge intelligence systems. In this section, we will first present several of those machine learning frameworks used across the machine learning domain and the developments that have adapted them to the more specific subdomain of edge intelligence. Then we will discuss considerations that users must take when selecting tools or frameworks for modeling and implementing edge intelligence in their own systems. Lastly, we will present a strategy for users to developing, implementing, and iterating on their machine learning design.

Machine Learning Frameworks for Edge Intelligence

The pervasive research, development, and usage of artificial intelligence and machine learning have yielded several machine learning frameworks, many with comprehensive sets of models and features, consistent use and recognition in academia and industry, and strong developer and community support. Many of the full-featured frameworks were originally developed for the more powerful hardware and platforms of the cloud computing paradigm; while these frameworks have matured significantly in terms of performance and efficiency, many are still not plausible for use on the kind of constrained hardware typically available on the network edge. To this end, specialty frameworks and extensions to existing frameworks have been developed to address the challenges posed by edge intelligence. When considering these frameworks, one should consider what hardware, architecture, and operating system platforms the framework requires, ease of development and deployment, availability of model implementations and implementation depth, diagnostics and visualization capabilities, and other specializations for edge devices (e.g., lightweight models, memory and computation optimizations, networking, and distributed infrastructure, etc.). Below, we explore two of the more popular machine learning frameworks and discuss their affinity for use on the network edge.

TensorFlow and TensorFlow Lite TensorFlow is an open-source deep learning framework created internally and managed by Google and made available for wider use in 2015 [24]. It is one of the more popular deep learning frameworks, used heavily in the domains of education, research, and industry. The standard, full-featured TensorFlow is available on Linux, Windows, and MacOS running on x86 platforms (with support for some ARM platforms) and can be implemented using Python, C++, and Java. It provides both high-level APIs for quick implementation and development of a wide range of machine learning models and low-level APIs for deep and involved implementation of deep learning models. Along with deep learning models and components, it also provides some distributed learning paradigms, such as built-in implementations of federated learning. TensorFlow can utilize diagnostics and visualization tools such as the TensorBoard suite of tools or the built-in profiler that can track memory and processing performance of TensorFlow models and operations. While the environment and computational requirements of Tensorflow may not suit the constraints of edge intelligence, the constraint-focused framework TensorFlow Lite allows users to take new or existing machine learning models, convert and compress them, deploy those models in constrained environments, and optimize them to run efficiently. TensorFlow Lite is optimized for on-device machine learning to address many of the key constraints of edge intelligence—including latency requirements, privacy, volatile and limited connectivity, model size and complexity, and power efficiency—by providing tools for model optimization and hardware acceleration [25]. To address the heterogeneous landscape of edge devices and environments, Tensorflow Lite supports a variety of platforms found on the edge—including Android, iOS, embedded Linux, and microcontrollers—as well as a variety of programming languages, including Java, Swift, Objective-C, C++, and Python. Lite's optimized model conversion covers the most common model types, but full optimization and acceleration do not cover all model types. The vast developer and community support for TensorFlow and TensorFlow Lite, including an abundance of documentation and end-to-end examples for a variety of applications, creates a vibrant and rich environment for development and experimentation. With TensorFlow's powerful and versatile model implementation features, growing support for distributed deep learning paradigms, and TensorFlow Lite's focus on and adaptations for on-device machine learning, TensorFlow and TensorFlow Lite can provide high- and low-level options for developing and deploying machine learning at the edge.

PyTorch and PyTorch Mobile PyTorch is another open-source deep learning framework originally developed by Meta AI and released in 2016 [26]. For most intents and purposes of edge intelligence system modeling and simulation, it is similar to TensorFlow, both in capability and positionality, and is also used heavily in the domains of education, research, and industry. It is developed for and available on Linux, Mac, and Windows and is fully available on both x86 and ARM architectures, in contrast to TensorFlow, which has relatively limited architecture support. Like TensorFlow, PyTorch provides both high-level APIs for abstracted development and low-level APIs for experimentation and fine-tuning, as well as some distributed and parallel training schemes. PyTorch comes with its own profiler for debugging models and traces, and with TorchViz, the visualization suite for PyTorch, and it can also be used with Tensorboard for more debugging and visualization features. Due to its native compatibility with the wide range of x86 and ARM machines running on popular operating systems, it is an excellent framework to use within portable environments, such as Docker, during simulation and experimentation. For deployment onto actual edge devices such as mobile phones and embedded systems, PyTorch provides PyTorch Mobile for Android, iOS, and Linux, which allows for low-latency and privacy-preserving machine learning model execution and encourages versatility and efficiency through the provision of optimized preprocessing and integration tools [27]. PyTorch has sizable, though not as vast as TensorFlow, developer and community support as well as effective documentation and examples across a range of applications, allowing for ease of development. With respect to TensorFlow and Keras, PyTorch models usually execute faster on most platforms due to their robust compatibility layers [28], but they lack the same level of memory, deployment, and quality-of-life optimizations that make TensorFlow the more popular option for developers in general. In any case, PyTorch is a powerful framework for developers of edge intelligence systems, with its ease of experimentation and development, low-level flexibility and versatility, and native availability on a wide range of operating systems and architecture platforms.

As more innovation and implementation go into the development of edge intelligence systems, more specialized machine learning tools and libraries are being developed, while mainstream machine learning tools are getting more efficient and powerful.

Data Modeling, Storage, and Preprocessing

In Sect. 2.2, we discussed the considerations and challenges of modeling edge intelligence system scenarios through data—including ensuring the adequate collection of good quality and useful data. We also discussed that storage, cleaning, and preparation are necessary to take raw data collected from sources and make it useful to the machine learning models driving our edge intelligence systems. We refer to this process of collecting and storing, cleaning, and standardizing, and preparing and assembling data before using them as inputs to machine learning models as data preprocessing.

1. Data must be collected and stored in file structures until processing, then loaded into data structures to facilitate data manipulation, i.e., the operations necessary for preprocessing.

2. Data then needs to be cleaned and corrected through the careful removal of noise, outliers, or irregular variance; adaptations to missing or erroneous data; and transformation and scaling.

3. Data must finally be prepared (e.g., encoded or restructured) and assembled into feature and label sets for training and testing.

Various methods of implementation are used at every stage of data preprocessing and while these methods can be implemented from scratch, many machine learning applications—and therefore, edge intelligence applications—share preprocessing methods, necessitating the development of various tools and platforms that can provide implementations for these methods. The machine learning frameworks we discussed in the previous section, such as TensorFlow and PyTorch, also include APIs for data storage, manipulation, and assembly. In addition to those tools, we also introduce some other tools and platforms that are useful for data storage, management, and preprocessing below.

Data Storage and Management In most modeling and experimental settings, it is common that you will be interfacing with static datasets, usually in the form of tabular data files for universal accessibility—e.g., CSV and TSV files—or files packaged for use with certain tools—e.g., TensorFlow or PyTorch datasets), which are essentially multidimensional data structures wrapped in manicured interfaces for use with compatible libraries. In some cases, you may even choose to process raw data from the source during runtime; e.g., directly inputting static text files to be processed and assembled into sequences of tokens for next word prediction models. Depending on what your design for the final product is, this approach may not be an efficient option, as the reprocessing of unchanged data for each run of your system would be unnecessarily redundant.

In other modeling and experimental settings and in actual deployment settings, data may be accessed through organized databases that live on storage devices either standalone or attached to your other end devices, e.g., the included HDD or SSD on a workstation or an SD card on a microcontroller. These databases can be accessed directly or through *database management systems* (DBMS). While DBMSs provide many powerful features for access, management, safety, and efficiency, the core purpose of modeling and simulating edge intelligence systems is to provide structured access and management. Different DBMSs can be used with different types of databases for different purposes and with different querying tools. Relational databases store data in tabular form that can be accessed via Structured Query Language (SQL). Object-oriented databases store data in the form of objects that work in the same way as objects in object-oriented programming languages and can be easily accessed via those languages. Non-relational databases do not follow strict formats like relational and object-oriented databases and provide flexibility in structure, which may incur some additional complexity when accessing data. These databases and their associated domain-specific languages are supported by most programming languages and platforms that would be used in the context of building edge intelligence systems. For example, tools like Pandas—a library that we discuss below—provide methods for extracting data from different types of databases and storing that data in objects within their own platform.

Data Analysis and Manipulation with Pandas Pandas is a data analysis and manipulation library built for Python, and its implementations are widely used and supported by many higher-level deep learning frameworks like TensorFlow and PyTorch and visualization frameworks like Matplotlib [29]. Pandas uses objects like its *DataFrames* and *Series* to store data in the form of tables and arrays. These objects are built atop *Numpy*, a popular and fundamental platform of scientific computing that provides multidimensional array objects and fast and efficient operations for those objects [30]. Accordingly, Pandas uses this functionality provided by Numpy to allow for scalable data manipulation, including selection, augmentation, and transformation of entire datasets, which is essential for processing data to be used as input to machine learning frameworks. Alternatives to using Pandas do exist, as a subset of its features can be implemented with other tools such as SciPy [31] and within Numpy. However, Pandas provides one of the most popular, widely used, and all-inclusive suites of data analysis tools for development and deployment of machine learning systems, with the added benefit of strong developer support and active development communities, making it easier to develop data solutions for these systems.

4.2 Network Simulation, Emulation, and Evaluation

Simulation and evaluation of computer networks already have a solid foundation and presence, given the extensive landscape of network simulators, network emulators, and network diagnostics tools. With the emergence of Internet-of-Things systems, many of these utilities have been extended with features that cover more complex network characteristics and behaviors, node or device characteristics and behaviors, and relevant hardware models and communication protocols. Some of these utilities, like network simulators, are standalone platforms that are feature-rich and typically include network configuration and performance evaluation tools, while requiring users to build desired application behaviors into the simulator environment. Other utilities, like network emulators, emulate just the network and are intended to be used alongside other system components. As we explore this landscape of utilities for network simulation, evaluation, and diagnostics, we also consider which of these options—and what they include and what they lack—works best for each of the scenarios that researchers and developers encounter.

Hardware Test Beds The best way to accurately and comprehensively observe and evaluate any system, including edge intelligent systems, in its target operating environment, is to build and deploy the system in that actual target environment. However, as noted before, building and deploying systems and their communication networks in their real environments is both time- and cost-intensive, and thus not an efficient first step to test and evaluate systems as they are being developed and iterated on. Hardware test beds are considered the next best solution for testing and validating systems in realistic environments without fully incurring high costs for testing and development. Hardware test beds utilize real hardware devices—whose parameters, behaviors, and protocols are often configurable by the user—in controlled or simulated environments such as server rooms or specialized chambers to minimize interference and allow full configurability [32]. Typically, users are required to have remote or local access to these test beds, set up their test environments and test devices with the necessary software or scenarios, and then deploy their experiments. While this process follows generally among network hardware test beds, the actual implementations of these test environments differ from one to another. In addition to the underlying communication network, hardware test beds can be used to test, validate, and evaluate the complete edge intelligence system, including end-user devices and applications (including machine intelligence) that run in the devices.

Because of their use of actual network hardware, including network devices and transmission media, hardware test beds are recognized as one of the more effective and accurate options for modeling and evaluating networks. However, similarly to deploying edge intelligence systems in their real operating environments, hardware test beds are not as accessible as other options for testing and evaluating edge intelligence systems. Performant hardware test beds can be expensive and not easily accessible. More recently, however, there has been an effort to make hardware test beds more available by allowing them to be remotely accessible and configurable to an extent for end-users, such as PAWR [33]. However, it is important to note that the user’s choice of devices, network, and configuration is limited by the test bed’s implementation and configuration. In addition, because of the dependence on real hardware devices and test bed setup and configuration, edge intelligence systems deployed in hardware test beds are not easily portable from one physical development space to another, as would be the case with network simulator- and emulator-based network models.

Network Emulators As discussed above, hardware test beds offer a solution to avoid the high costs of deploying real devices and the underlying communication infrastructure in the target operating environments by deploying them in configurable and controlled test environments. Network emulators, on the other hand, connect real or virtualized end devices or systems with emulated networks, thus further simplifying deployment, reducing costs, and increasing accessibility and portability. Network emulators are implemented as software that typically runs on single machines that aim to emulate the behaviors—including transmission media characteristics and network stack protocols—of the target network. Network emulators provide interfaces that real or virtualized end devices or systems can use to connect to and utilize the emulated network as they would with a real network. While some network emulators do come with built-in or proprietary diagnostic tools, the use of real end-user devices, allows for the use of end-to-end network diagnostic tools included in the host operating system, such as the *ping* and *iperf* tools [34]. One notable network emulator is Mininet [34, 35], along with its variants, including Mininet-Wifi [36] and Distri-net [37].

Network emulators can be thought of as occupying an intermediate space between hardware test beds and network simulators (which will be described in detail below). Like hardware test beds, they use real or virtualized end devices or systems, but use emulated networks and network devices in lieu of real networks and network devices, a common feature they share with network simulators. This makes network emulators attractive when developers are building services and applications, have access to real end devices, and would like to observe and test their performance in a test network before deploying it in hardware test beds or real networks.

While the distinction between network emulators and simulators is marked by whether devices interface with emulated networks or are simulated within the simulated network, many network simulator and emulator platforms can be used both as simulators and emulators. For example, Mininet allows you to use simulated host devices and systems to connect over its emulated network, thus effectively making it behave as a network simulator. On the other hand, network simulators like *OMNET++* and *ns-3* have "emulation modes" that allow real or virtual end devices or systems to interface with the simulated network through direct code execution [38] or TAP interfaces [39].

Network Simulators Network simulators are powerful, feature-rich software tools that allow users to build, configure, and deploy simulated networked systems. While the depth and complexity of the network simulation depend on the network simulator itself, most state-of-the-art network simulators come with features for modeling and simulating elements and modules across the network stack, including the transmission media, data and packet encapsulation and decapsulation, communication protocols, and even edge device behaviors such as mobility and resource utilization. Network simulators have numerous benefits, over other options for modeling and simulating edge networks and edge intelligent systems. We list some of these benefits below:

- The wide range of features and implementations available in state-of-the-art network simulators facilitate modeling the rich landscape of edge networks and devices with adequate fidelity.
- The ability to configure the simulated networks and deploy them with user-programmed scenarios in a controlled environment allows for better

modeling of network behavior, versatile testability, and experimental reproducibility.
- Network simulators’ accessibility and cost-effectiveness (with reference to resource costs, time costs, and financial costs) allow for faster development of working models, better scalability with regard to the size and complexity of simulated networks, and, as a result, more widespread use in industrial, research, and educational settings.
- Additionally, network simulators’ rich feature sets and controlled environments make possible a capable suite of network diagnostic tools to observe, monitor, and evaluate the simulated networks, allowing users to infer and predict the results of their deployments.

Some of the more well-known network simulators used in research, industry, and education include *OMNET++* [13] and *ns-3* [15]. While ns-3 and OMNET++ share many similarities and differences, both of them are effective options to model and evaluate edge networks. Both are discrete-event simulators, allowing for controlled and comprehensive network event processing organized into discrete time steps, which is especially important when modeling and observing network activity in the potentially dynamic and complex environments present in edge networks. Both simulators also offer basic models of network devices (servers, clients, antennas, etc.), basic transmission media (wired, wireless, etc.), and protocols, allowing users to build and test a variety of network scenarios. Some network simulators offer detailed models that capture the characteristics and behaviors of specific network devices and communication protocols; ns-3 offers various such module implementations built into its core libraries, whereas OMNET++ provides the *Inet* [14] add-on framework to make available a wide range of real-world network devices, protocols, and scenario templates. OMNET++, which offers both commercial and academic solutions, includes a full-feature custom IDE, GUIs for setting network design and configuration parameters, and visualization software that offers deep control of scenario execution and reporting. Meanwhile, ns-3 offers more comprehensive modeling capabilities, allowing experienced users to work with and extend deeper existing models, resulting in a more powerful network simulation experience. While both simulators use C++ as the primary programming language to implement network and component models—thus making them readily accessible to the of C++ programmer community at large—ns-3 also allows the use of Python instead of C++ to implement network models—thus catering to the large, growing community of Python programmers. Both simulators are well-documented with tutorials, libraries, and technical documentation, enriched through contributions from their large user communities, and are maintained, updated, and extended by highly responsive development teams, ensuring that the simulators evolve with emerging paradigms, models, and protocols, including edge systems.

Aside from these all-purpose network simulators, there also exist more niche, scenario- or deployment-specific network simulators. For example, *Cooja* [40], is a simulator designed to simulate wireless sensor networks and IoT environments. Cooja simulates nodes that run the Contiki operating system ContikiOS and emulates a variety of hardware platforms. Another example is Cisco's *NetSim* [41], which is an education-oriented network and router simulator intended for use with commercial equipment and network infrastructure but has been adopted for IoT and edge applications. *IoTify* is a network simulation platform that is targeted at modeling networks and digital twinning for large-scale industrial IoT deployments [42].

While network simulators can be effective tools for modeling edge intelligence systems' underlying networks, users should also consider their shortcomings. Unlike network emulators, which are functional proxies for the real network, network simulators are usually standalone, self-contained platforms, and therefore application and device behaviors often have to be built atop the simulator's custom framework and environment. As mentioned before, while "emulation modes" do exist for some network simulators, they are implemented using operating system-specific tools like Linux's TAP interface and are therefore not a universally accessible option. Unlike hardware test beds, network simulators cannot always or truly represent the actual, real-world hardware and physical environment at a level of fidelity that captures their exact behaviors. Similarly, unlike actual real-world deployment, network simulators cannot truly model all the unpredictable elements that exist in the environments in which real networks function. Lastly, network simulators may not be able to model true real-time networks, as all of their components use simulated time and are therefore unable to model the true timing of network events.

Custom Network Simulation with Functional Parameters Sometimes, modeling the network component by component may be impossible—due to limitations in the available development tools, environments, or processes—or unnecessary, depending on the needs of your modeling, simulating, and evaluation. In these cases, it may then be necessary or easier to reduce our network component-based model to a simpler functional network model. This is especially useful for development in the absence of or lack of access to more capable network simulators. The implementation of this simple functional network model can vary depending on the needs of the researcher or developer and the development environment. Despite these differences, there are a few features and metrics that are commonly considered, including, but not limited to, the following:

- *Packet Loss.* The rate at which messages might be dropped or lost can be represented by a simple probability model. However, while this is simple to model, packet loss in the real world is not a true heuristic but rather the result of other network performance features.
- *Latency.* How long it takes for a message to travel from node to node can be represented by implementing delays. The length of these delays can be arbitrary or based on the configuration of the network and the path of the message. These delays can be implemented as a non-blocking delay based on the system clock or, in the case of non-real time implementations, as an increment to the system counter.
- *Transmission Range.* The range in which a node can send and receive messages to and from another node can be implemented through a conditional statement that measures the Euclidean distance between two location coordinates and then decides whether the message transmits successfully or not. Whether the message transmits can be decided either by simply observing whether or not the involved nodes are within range of each other or through more detailed implementations that consider RSSI and/or other preset parameters or function literals. Determining this range is a question of connectability and reachability and can be affected by objects interfering with signals and the mobility of the nodes themselves.
  - *Signal interference.* Though signal interference is more often omitted from simpler models, in the real world, physical objects can block,

weaken, or scatter radio signals, affecting the connectivity between nodes.

- *Mobility*. The mobility of network nodes—or objects that cause interference—affects the position of the nodes or objects and therefore can affect whether two nodes are within range of each other. Mobility patterns are implemented as features in some network simulators, but they can also be implemented as functions that update the node's or object's position over time.

4.3 Runtime Environments

Edge intelligence systems are, by their nature, distributed and likely heterogeneous, consisting of all the different components that make up the infrastructure and function of the system. How do we model and simulate such a system with different functions executed by different programs running on different platforms on different devices, all of which could be connected via different networks? The immediately obvious solution is to build the system as it was designed, complete with all its infrastructure and functional components, including the physical hardware and network. However, due to cost and time constraints, this solution is not always affordable or feasible. Building and deploying systems with all their components, connections, and configurations can be cost-intensive and time-intensive, making iteration and development burdensome or unaffordable, if not impossible.

Full Virtualization The go-to solution to this challenge has been *virtualization* [43] in which hardware, firmware, or software can be emulated by abstracting these components away from their underlying resources so that a virtualized element can be spun up and deployed within a host environment. Full hardware virtualization is the full emulation of underlying hardware resources such that software built for the target hardware would be able to run without issue on the virtualized hardware, whereas paravirtualization does not emulate the hardware but still provides a space separate from the host system. In the context of edge intelligence systems, with full hardware emulation, it is theoretically possible to virtualize the infrastructure and platform components—the hardware and operating systems—and run the functional components—the software—atop the virtualized infrastructure. The use of virtual machines (VMs) with full hardware virtualization through VM management tools like *VirtualBox* [44] or *VMWare* [45] enables researchers and developers to virtualize not just the hardware and operating systems but also other infrastructure components, including network components, storage devices, and interfaces. While full virtualization can be a potent solution for some systems without tight constraints, the resource overhead incurred by spinning up hardware virtualizations and the potential incompatibility across hardware platforms can be prohibitive for developing larger and more distributed systems. Fortunately, full hardware and operating system virtualization is not always necessary for testing and development purposes.

Containerization Containerization is a form of paravirtualization in which the software environment and components are virtualized such that each *container*—the self-contained environment and its software—is separate from the other and from the host system. With edge intelligence systems, this enables the execution of many of the functional components within containers representing the different end devices and the simulation of some hardware features like processing, memory, and storage availability, as well as some network features [46]. The use of containerization tools like *Docker* and container cluster orchestration tools like *Kubernetes* has become more pervasive as research and development of connected systems and microservices have become more relevant [47]. The ability to virtualize key aspects of the functional components in a distributed manner enables scalable simulation of larger, more distributed, and more heterogeneous systems [46, 47]. However, in the context of edge intelligence systems, the lack of hardware virtualization does introduce some limitations to being able to simulate the full range or depth of infrastructure components, and containerized system simulations may not always be portable from one operating system or hardware platform to another.

Let us consider a case study of integrating the popular deep learning framework, TensorFlow, into a simulated model of an edge intelligence system. For this case, it is important to know that TensorFlow does not natively provide support for ARM architecture platforms and is built and compiled for the x86 architecture. Suppose that we use an x86-based Windows machine to initially develop our simulated model of a TensorFlow-enabled edge intelligence system as a containerized system in which each device is represented as a container running their respective functions and all devices are connected via the Docker bridge network. Suppose that we want to port our simulated model to our colleagues using ARM-based Mac machines. Because TensorFlow does not provide ARM support and because Docker does not provide instruction-level virtualization, the simulated model would not be able to execute on the colleagues’ machines. In addition to that, our actual design for the edge intelligence system may incorporate ARM-based devices running low-level firmware processes that could not be simulated properly on x86-based containers. Due to the limitations of full hardware virtualization and depending on the platforms used for development, it may not be possible even with virtualization. At the time of writing, Docker for ARM-based Mac machines supports some level of instruction-level translation to service x86-based software through the Rosetta 2 translation service. TensorFlow may release a version of its full suite in which all tools and operations are fully executable on ARM-based machines. However, these solutions are outside the scope of containerization, so it is still critical to exercise caution and careful consideration about how systems can be developed, ported, and deployed through virtualization, whether it be with full hardware virtualization or containerization.

---

5 End-to-End and Integrated Solutions for Simulation

While component solutions to modeling and simulation provide valuable insights into design and performance, edge intelligence systems are inherently complex. In other words, the modeling and simulation of edge intelligence systems go beyond the modeling and simulation of individual components and must consider the interactivity between those components. Ideally, for the purposes of consistency and interoperability between components, the best solutions for modeling and simulation would be all-inclusive with sufficient flexibility and modularity to accommodate the diversity of edge intelligence system design. However, this does not reflect the current state of edge intelligence system simulators, which have largely diverged into specialized end-to-end simulators and general purpose integrations of component simulations.

As the emergence of edge intelligence synthesized around the growth and coalescence of different domains [2]—namely edge computing, distributed machine learning, and distributed systems—the current state of edge intelligence simulation is in part served by more specialized simulators that each prioritize solving the challenges presented in some of these domains over others. The need to model and simulate resource management and allocation strategies found solutions in simulators like *SimEdgeIntel* [48]—a highly configurable and accessible simulator built to model important aspects of networking environments with machine learning—and *LoRa-MAB* [49]—a LoRa network-specific simulator that can model and execute distributed reinforcement learning algorithms for resource allocation. The need to model and evaluate distributed machine learning algorithms in the edge context has largely been addressed through tools and implementations provided by popular deep learning frameworks, such as federated learning implementations in TensorFlow or distributed training implementations in PyTorch. These implementations have been widely used in virtualized or containerized environments to simulate truly distributed learning environments and paradigms.

However, the lack of general purpose and modular simulation options for edge intelligence systems has created a vacuum that is being filled with solutions that integrate component simulation solutions together. The mentioned solutions in which learning algorithms are deployed in containers, are one such example of integrating distributed learning algorithms with distributed runtime environments. However, the capability of modeling the networking aspect is limited by the often-rudimentary network emulations provided by container management systems. To this end, solutions that enable building edge intelligence applications atop research-oriented network simulators have emerged to utilize the depth of simulation offered by popular network simulators. *Dockemu* [50] is one such simulation tool that allows users to deploy any application—including machine learning applications—within Docker containers and simulate the networking through the ns-3 network simulator via the utilization of Linux TAP interfaces to connect containers through the simulated network. To more directly enable machine learning experimentation on simulated networks, the *ns3-ai* module [51] built for the ns-3 network simulator provides the ability to connect machine learning processes to simulated networks using shared memory between the network simulation component and the machine learning components. While these solutions are built specifically for use with the ns-3 network simulator, there are similar implementations built for other network simulators, such as OMNET++, built with universally accessible tools like TensorFlow APIs or *OpenAI Gym* [52]. Beyond tools built for specific applications, libraries, and network simulators, *NSB (Network Simulation Bridge)* [53] provides a simulator-agnostic and platform-agnostic tool to connect application front-ends to simulated network back-ends through the use of high-level APIs and low-level message parsing interfaces to communicate with a distributed event processing daemon to relay communication between front-end agents over the back-end simulated network.

While all these solutions address the end-to-end modeling, simulation, and evaluation needs in some form, there continues to be a need for more full-fledged simulation solutions that can be sufficiently accessible, capable, and complex enough to enable exploration of these systems. This need for adequate experimental infrastructure is identified as an open problem in the continued development of edge intelligence as a domain [2].

---

6 Conclusion

The growing landscape of connected devices and the increasing capabilities of those devices and the infrastructure that connects them have created rich opportunities for research and development at the network edge. The proliferation of machine learning being implemented into applications ranging from industrial and infrastructure applications to everyday consumer applications has benefited from this growth and investment into the network edge through the increasing amount of data and computational ability at the edge. While this symbiotic relationship has enabled the emergence of edge intelligence and the development of systems that are built on edge intelligence, the modeling, simulation, and evaluation of edge intelligence systems is still a maturing discipline. In this chapter, we introduce concepts, strategies, and tools to enable researchers and developers to find modeling and simulation solutions so that they can better design and evaluate their edge intelligence systems.

We acknowledge that while what is offered in this chapter—and more broadly, what is offered in the state-of-the-art—provides comprehensive solutions for modeling, simulation, and evaluation of components derived from networking, edge computing, and machine learning, they may not provide complete and comprehensive solutions to the modeling, simulation, and evaluation of all edge intelligence systems. It is important to understand that the current landscape of these solutions reflects the newness of the edge intelligence space. As the development of such systems becomes more ubiquitous, today's state-of-the-art could form the foundation for more powerful solutions that enable better creativity, versatility, and fidelity for the edge intelligence systems of tomorrow.

References

1. Sharma, N., Shamkuwar, M., Singh, I.: The history, present and future with IoT. In: Balas, V.E., Solanki, V.K., Kumar, R., Khari, M. (eds.) Internet of Things and Big Data Analytics for Smart Generation, pp. 27–51. Springer, Cham (2019)
   [Crossref]
2. Deng, S., Zhao, H., Fang, W., Yin, J., Dustdar, S., Zomaya, A.Y.: Edge intelligence: the confluence of edge computing and artificial intelligence. IEEE Internet Things J. **7**, 7457–7469 (2020). https://doi.org/10.1109/JIOT.2020.2984887
   [Crossref]
3. Xu, D., Li, T., Li, Y., Su, X., Tarkoma, S., Jiang, T., Crowcroft, J., Hui, P.: Edge Intelligence: Architectures, Challenges, and Applications, http://arxiv.org/abs/2003.12172 (2020)
4. Mendez, J., Bierzynski, K., Cuéllar, M.P., Morales, D.P.: Edge intelligence: concepts, architectures, applications, and future directions. ACM Trans. Embed. Comput. Syst. **21**, 41–48 (2022). https://doi.org/10.1145/3486674
   [Crossref]
5. Rayhana, R., Xiao, G., Liu, Z.: Internet of things empowered smart greenhouse farming. IEEE J. Radio Freq. Identif. **4**, 195–211 (2020). https://doi.org/10.1109/JRFID.2020.2984391
   [Crossref]
6. McMahan, B., Moore, E., Ramage, D., Hampson, S., Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, pp. 1273–1282. PMLR (2017)
7. Hoffpauir, K., Simmons, J., Schmidt, N., Pittala, R., Briggs, I., Makani, S., Jararweh, Y.: A survey on edge intelligence and lightweight machine learning support for future applications and services. J. Data Inf. Qual. **15**, 1–20.: 30 (2023). https://doi.org/10.1145/3581759
   [Crossref]
8. Zebari, R., Abdulazeez, A., Zeebaree, D., Zebari, D., Saeed, J.: A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction. J. Appl. Sci. Technol. Trends. **1**, 56–70 (2020). https://doi.org/10.38094/jastt1224
   [Crossref]

9. Shorten, C., Khoshgoftaar, T.M., Furht, B.: Text data augmentation for deep learning. J. Big Data. **8**, 101 (2021). https://doi.org/10.1186/s40537-021-00492-0
[Crossref]

10. Shorten, C., Khoshgoftaar, T.M.: A survey on image data augmentation for deep learning. J. Big Data. **6**, 1–48 (2019). https://doi.org/10.1186/s40537-019-0197-0
[Crossref]

11. Vabalas, A., Gowen, E., Poliakoff, E., Casson, A.J.: Machine learning algorithm validation with a limited sample size. PLoS One. **14**, e0224365 (2019). https://doi.org/10.1371/journal.pone.0224365
[Crossref]

12. de Carvalho Silva, J., Rodrigues, J.J.P.C., Alberti, A.M., Solic, P., Aquino, A.L.L.: LoRaWAN – a low power WAN protocol for Internet of Things: a review and opportunities. In: 2017 2nd International Multidisciplinary Conference on Computer and Energy Science (SpliTech), pp. 1–6 (2017)

13. Varga, A., Hornig, R.: An overview of the OMNeT++ simulation environment. Presented at the 1st International ICST Conference on Simulation Tools and Techniques for Communications, Networks and Systems, May 16 (2010)

14. Mészáros, L., Varga, A., Kirsche, M.: INET framework. In: Virdis, A., Kirsche, M. (eds.) Recent Advances in Network Simulation: The OMNeT++ Environment and its Ecosystem, pp. 55–106. Springer, Cham (2019)
[Crossref]

15. Riley, G.F., Henderson, T.R.: The ns-3 network simulator. In: Wehrle, K., Güneş, M., Gross, J. (eds.) Modeling and Tools for Network Simulation, pp. 15–34. Springer, Berlin/Heidelberg (2010)
[Crossref]

16. Shakkottai, S., Rappaport, T.S., Karlsson, P.C.: Cross-layer design for wireless networks. IEEE Commun. Mag. **41**, 74–80 (2003). https://doi.org/10.1109/MCOM.2003.1235598
[Crossref]

17. Aslanpour, M.S., Gill, S.S., Toosi, A.N.: Performance evaluation metrics for cloud, fog and edge computing: a review, taxonomy, benchmarks and standards for future research. Internet of Things. **12**, 100273 (2020). https://doi.org/10.1016/j.iot.2020.100273
[Crossref]

18. Murshed, M.G.S., Murphy, C., Hou, D., Khan, N., Ananthanarayanan, G., Hussain, F.: Machine learning at the network edge: a survey. ACM Comput. Surv. **54**, 1–170.:37 (2021). https://doi.org/10.1145/3469029
[Crossref]

19. Merluzzi, M., Lorenzo, P.D., Barbarossa, S.: Wireless edge machine learning: resource allocation and trade-offs. IEEE Access. **9**, 45377–45398 (2021). https://doi.org/10.1109/ACCESS.2021.3066559
[Crossref]

20. Chen, M., Gündüz, D., Huang, K., Saad, W., Bennis, M., Feljan, A.V., Poor, H.V.: Distributed learning in wireless networks: recent progress and future challenges. IEEE J Sel Areas Commun. **39**, 3579–3605 (2021). https://doi.org/10.1109/JSAC.2021.3118346
[Crossref]

21. Mills, J., Hu, J., Min, G.: Communication-efficient federated learning for wireless edge intelligence in IoT. IEEE Internet Things J. **7**, 5986–5994 (2020). https://doi.org/10.1109/JIOT.2019.2956615
[Crossref]

22. Menghani, G.: Efficient deep learning: a survey on making deep learning models smaller, faster, and better. ACM Comput. Surv. **55**, 1–259.:37 (2023). https://doi.org/10.1145/3578938
[Crossref]

23. Jiang, C., Cheng, X., Gao, H., Zhou, X., Wan, J.: Toward computation offloading in edge computing: a survey. IEEE Access. **7**, 131543–131558 (2019). https://doi.org/10.1109/ACCESS.2019.2938660
[Crossref]

24. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D.G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., Zheng, X.: {TensorFlow}: a system for {Large-Scale} machine learning. Presented at the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16) (2016)

25. Deng, Y.: Deep learning on mobile devices: a review. In: Mobile Multimedia/Image Processing, Security, and Applications 2019, pp. 52–66. SPIE (2019)

26. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: PyTorch: an imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems. Curran Associates (2019)

27. Luo, C., He, X., Zhan, J., Wang, L., Gao, W., Dai, J.: Comparison and Benchmarking of AI Models and Frameworks on Mobile Devices, http://arxiv.org/abs/2005.05085 (2020)

28. Vasilev, I., Slater, D., Spacagna, G., Roelants, P., Zocca, V.: Python Deep Learning: Exploring Deep Learning Techniques and Neural Network Architectures with PyTorch, Keras, and TensorFlow, 2nd edn. Packt Publishing Ltd (2019)

29. Hunter, J.D.: Matplotlib: a 2D graphics environment. Comput. Sci. Eng. **9**, 90–95 (2007). https://doi.org/10.1109/MCSE.2007.55
[Crossref]

30. van der Walt, S., Colbert, S.C., Varoquaux, G.: The NumPy array: a structure for efficient numerical computation. Comput. Sci. Eng. **13**, 22–30 (2011). https://doi.org/10.1109/MCSE.2011.37
[Crossref]

31. Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, İ., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P.: SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat. Methods. **17**, 261–272 (2020). https://doi.org/10.1038/s41592-019-0686-2
[Crossref]

32. Pakzad, F., Portmann, M., Turletti, T., Parmentelat, T., Mahfoudi, M.N., Dabbous, W.: R2Lab testbed evaluation for wireless mesh network experiments. In: 2018 28th International Telecommunication Networks and Applications Conference (ITNAC), pp. 1–6 (2018)

33. Gosain, A.: Platforms for advanced wireless research: helping define a new edge computing paradigm. In: Proceedings of the 2018 on Technologies for the Wireless Edge Workshop, p. 33. Association for Computing Machinery, New York (2018)

34. Bholebawa, I.Z., Jha, R.K., Dalal, U.D.: Performance analysis of proposed OpenFlow-based network architecture using Mininet. Wirel. Pers. Commun. **86**, 943–958 (2016). https://doi.org/10.1007/s11277-015-2963-4
[Crossref]

35. de Oliveira, R.L.S., Schweitzer, C.M., Shinoda, A.A., Prete, L.R.: Using Mininet for emulation and prototyping software-defined networks. In: 2014 IEEE Colombian Conference on Communications and Computing (COLCOM), pp. 1–6 (2014)

36. Fontes, R.R., Afzal, S., Brito, S.H.B., Santos, M.A.S., Rothenberg, C.E.: Mininet-WiFi: emulating software-defined wireless networks. In: 2015 11th International Conference on Network and Service Management (CNSM), pp. 384–389 (2015)
[Crossref]

37. Di Lena, G., Tomassilli, A., Saucez, D., Giroire, F., Turletti, T., Lac, C.: Distrinet: a mininet implementation for the cloud. SIGCOMM Comput. Commun. Rev. **51**, 2–9 (2021). https://doi.org/10.1145/3457175.3457177
[Crossref]

38. Camara, D., Tazaki, H., Mancini, E., Turletti, T., Dabbous, W., Lacage, M.: DCE: test the real code of your protocols and applications over simulated networks. IEEE Commun. Mag. **52**, 104–110 (2014). https://doi.org/10.1109/MCOM.2014.6766093
[Crossref]

39. Fontes, H., Cardoso, T., Ricardo, M.: Improving ns-3 emulation performance for fast prototyping of network protocols. In: Proceedings of the 2016 Workshop on ns-3, pp. 108–115. Association for Computing Machinery, New York (2016)
[Crossref]

40. Osterlind, F., Dunkels, A., Eriksson, J., Finne, N., Voigt, T.: Cross-level sensor network simulation with COOJA. In: Proceedings. 2006 31st IEEE Conference on Local Computer Networks, pp. 641–648 (2006)
[Crossref]

41. Rathi, A.K., Santiago, A.J.: The new NETSIM simulation program. Traffic Eng. Control. **31** (1990)

42. Dias, J.P., Couto, F., Paiva, A.C.R., Ferreira, H.S.: A brief overview of existing tools for testing the internet-of-things. In: 2018 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), pp. 104–109 (2018)
[Crossref]

43. Mansouri, Y., Babar, M.A.: A review of edge computing: features and resource virtualization. J Parallel Distrib Comput. **150**, 155–183 (2021). https://doi.org/10.1016/j.jpdc.2020.12.015
[Crossref]

44. Watson, J.: Virtualbox: bits and bytes masquerading as machines. Linux J. **2008**, 1 (2008)

45. Bugnion, E., Devine, S., Rosenblum, M., Sugerman, J., Wang, E.Y.: Bringing virtualization to the x86 architecture with the original VMware workstation. ACM Trans. Comput. Syst. **30**, 1–12.:51 (2012). https://doi.org/10.1145/2382553.2382554
[Crossref]

46. Morabito, R.: Virtualization on internet of things edge devices with container technologies: a performance evaluation. IEEE Access. **5**, 8835–8850 (2017). https://doi.org/10.1109/ACCESS.2017.2704444
[Crossref]

47. Huang, Y., Cai, K., Zong, R., Mao, Y.: Design and implementation of an edge computing platform architecture using Docker and Kubernetes for machine learning. In: Proceedings of the 3rd International Conference on High Performance Compilation, Computing and Communications, pp. 29–32. Association for Computing Machinery, New York (2019)
[Crossref]

48. Wang, C., Li, R., Li, W., Qiu, C., Wang, X.: SimEdgeIntel: a open-source simulation platform for resource management in edge intelligence. J. Syst. Archit. **115**, 102016 (2021). https://doi.org/10.1016/j.sysarc.2021.102016
[Crossref]

49. Ta, D.-T., Khawam, K., Lahoud, S., Adjih, C., Martin, S.: LoRa-MAB: a flexible simulator for decentralized learning resource allocation in IoT networks. In: 2019 12th IFIP Wireless and Mobile Networking Conference (WMNC), pp. 55–62 (2019)
[Crossref]

50. Petersen, E., Cotto, G., Antonio, T.M.: Dockemu 2.0: evolution of a network emulation tool. In: 2019 IEEE 39th Central America and Panama Convention (CONCAPAN XXXIX), pp. 1–6 (2019)

51. Yin, H., Liu, P., Liu, K., Cao, L., Zhang, L., Gao, Y., Hei, X.: ns3-ai: fostering artificial intelligence algorithms for networking research. In: Proceedings of the 2020 Workshop on ns-3, pp. 57–64. Association for Computing Machinery, New York (2020)
[Crossref]

52. Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., Zaremba, W.: OpenAI Gym, http://arxiv.org/abs/1606.01540 (2016)

53. Kuttivelil, H.S., Sreenivasamurthy, S., Krishnaswamy, L., Bhatia, N., Obraczka, K.: Network simulation bridge: bridging applications to network simulators. In: Proceedings of the 19th ACM International Symposium on QoS and Security for Wireless and Mobile Networks, pp. 39–46. Association for Computing Machinery, New York (2023)
[Crossref]

OceanofPDF.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024

S. Pal et al. (eds.), *IoT Edge Intelligence*, Internet of Things: Technology, Communications and Computing
https://doi.org/10.1007/978-3-031-58388-9_2

Exploring Edge AI Inference in Heterogeneous Environments: Requirements, Challenges, and Solutions

Roberto Morabito[1] ✉ and Mung Chiang[2] ✉
(1) University of Helsinki, Helsinki, Finland
(2) Purdue University, West Lafayette, IN, USA

✉ **Roberto Morabito (Corresponding author)**
Email: roberto.morabito@helsinki.fi

✉ **Mung Chiang**
Email: chiang@purdue.edu

Keywords Edge Intelligence – Edge AI – Networked AI systems – AI inference – AI accelerators (GPU, TPU, VPU)

1 Introduction

Artificial intelligence (AI) has considerably changed the computer technology landscape in the last decade, with extensive impacts across various sectors, including industrial automation, automotive, eXtended Reality (XR), unmanned aerial vehicles (UAVs), and smart cities [1]. The integration of AI capabilities into these sectors often involves the use of sophisticated machine learning (ML) algorithms, including deep learning (DL) [2]. These algorithms typically require dedicated hardware platforms for executing ML inference tasks, setting aside the training phase, which typically occurs in more computationally capable environments.

ML inference refers to the process of using trained ML models to make predictions on new data. Unlike the training phase, which involves learning the model parameters from large datasets and is often computationally intensive, the inference phase is about using the trained models to make predictions in real time [3]. This capability is paramount for applications demanding stringent real-time processing requirements while fulfilling their Quality of Service (QoS) needs [4, 5].

Two major technological trends have emerged due to the quest for efficient ML inference execution:

(i) An escalating demand for boosting AI inference capabilities at the edge.

(ii) The development of innovative AI hardware and software solutions.

These trends are clearly reflected in the growing efforts of chipset manufacturers and software developers to create customized solutions that harness the full potential of edge computing powered by AI, often referred to as *Edge AI*. For instance, the development and market release of various AI accelerator chipsets like graphics processing units (GPUs), vision processing units (VPUs), and tensor processing units (TPUs) have been specifically designed to facilitate the execution of Edge AI inference tasks [6].

Interestingly, a significant portion of these Edge AI chips is deployed in consumer devices. According to [7], as of 2020, consumer devices constituted over 90% of the Edge AI chip market in terms of both units sold and dollar value. The same source indicates that high-end smartphones make up more than 70% of all consumer Edge AI chips currently in use, with tablets, wearables, and smart speakers also incorporating these chips. This scenario underscores the high heterogeneity and distributed nature of edge devices executing AI inference tasks.

Within this context, we explore the multifaceted aspects of AI inference provisioning at the edge, placing particular emphasis on complex deployment scenarios. These scenarios often require heterogeneous AI-enabled edge nodes to cooperate in order to perform and deliver QoS-compliant AI inference tasks. Such circumstances must account for the typically limited computational capabilities of edge devices, as well as the impact of network latency, especially when edge nodes process AI inference results over data received from end-devices (Fig. 1).

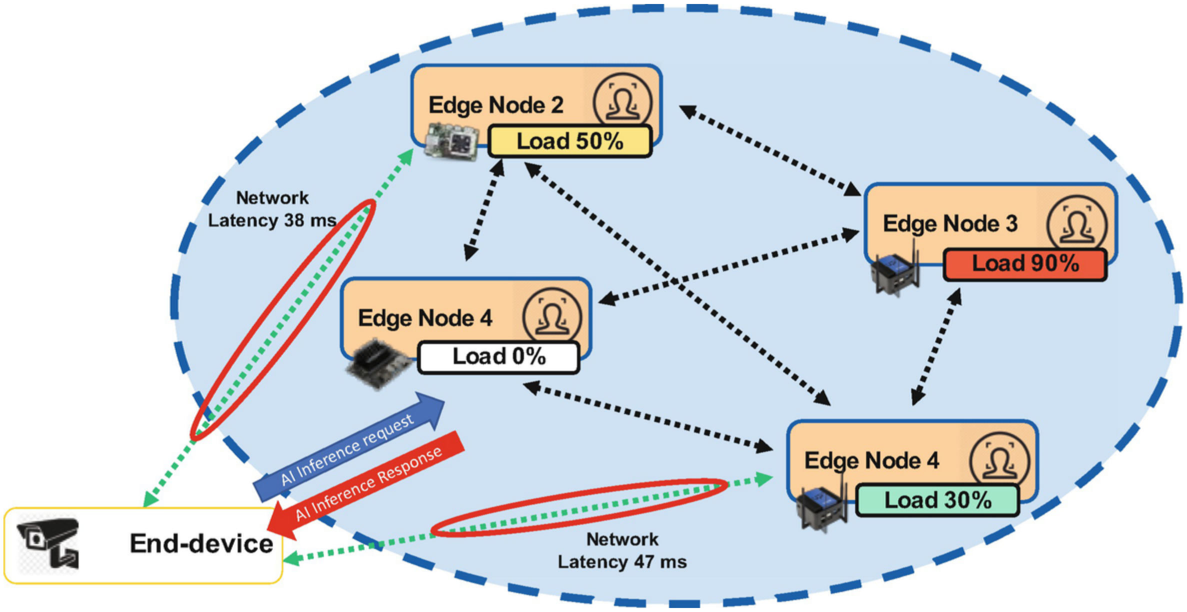

Edge Node 2
Load 50%
Edge Node 3
Load 90%
Edge Node 4
Load 0%
Edge Node 4
Load 30%
Network
Latency 38 ms
Network
Latency 47 ms
AI Inference request
AI Inference Response
End-device

Fig. 1 Cluster of heterogeneous AI-enabled edge devices, each experiencing different computational loads and varying degrees of network latency in relation to the end device that has initiated an inference request

Moreover, as the implementation of AI capabilities in various devices expands, and the deployment of complex Edge AI systems continues to grow, there is an increasing need for solutions that can seamlessly discover, provision, and orchestrate AI inference services across various, distributed edge nodes.

Against this backdrop, the core objective of this chapter is to introduce a framework that facilitates the seamless management of these heterogeneous, distributed Edge AI devices. Our work, a thorough extension and elaboration of our award-winning demo paper [8], aims to contribute a solution that can improve the management of future Edge AI systems. We propose an architecture that considers AI-enabled hardware diversity as a resource for next-generation edge computing systems. This is accomplished by shedding light on this domain, delving into some of the most notable research challenges in the field, and proposing a platform-agnostic architectural framework that ensures seamless discovery, provisioning, and orchestration of AI inference services across a network of heterogeneous and distributed edge nodes.

Before exploring the practical aspects of deploying distributed, heterogeneous Edge AI systems, it is essential to comprehend the broader concept of Edge Intelligence (EI), which encompasses Edge AI. Accordingly, we provide an in-depth examination of EI in Sect. 2. Section 3 shifts the discussion to heterogeneous Edge AI inference, outlining its potential benefits, addressing the unique combination of hardware and software involved, detailing the practical requirements and challenges of implementation, and exploring practical scenarios to provide readers with tangible insights. Section 4 introduces a framework designed to facilitate inference provisioning in the heterogeneous environments discussed in Sect. 3. This section describes the architecture of the framework, illustrates its application through various cases of inference provisioning, and demonstrates how it meets the previously discussed requirements and challenges. Finally, the chapter concludes with Sect. 5, summarizing the insights gained from the previous sections and suggesting directions for future research and development in the field of Edge AI inference within heterogeneous environments.

2 Understanding Edge Intelligence: An In-depth Look

As we transition from the introductory considerations, it is essential to understand the broader concept of EI before delving into the practical aspects of deploying distributed, heterogeneous Edge AI systems. To this end, this section provides an overview of EI, explaining what it entails, spotlighting practical instances of its application, and contextualizing Edge AI within this framework. By laying this foundation, we aim to emphasize the significance of EI in our increasingly interconnected digital ecosystem, thereby setting the stage for subsequent sections that explore the unique challenges inherent to EI and Edge AI inference within heterogeneous systems.

2.1 What Is Edge Intelligence

EI is an emerging concept in computational paradigms that represents the integration of edge computing and AI. Understanding these two components is key for a deeper insight into EI [9, 10].

Edge computing, an extension of cloud computing, brings cloud services closer to end-users. It provides computing, storage, and networking resources at the network's edge through mediums like IoT gateways, routers, microdata centers, base stations, etc.—often referred to as edge nodes or edge devices. Conversely, entities like mobile phones and IoT devices that seek services from these edge nodes are called end devices. In the rest of this chapter, we will use this terminology, although slightly different terms are used in [11]. Edge computing marks a significant shift from centralized cloud computing towards a more decentralized model, offering low latency due to the proximity of data processing to the source. It empowers end devices to offload computing tasks to edge nodes, conserving energy and enhancing device battery life [12].

The term "intelligence" usually refers to the ability to interact with the environment, set and achieve goals, and modify behavior in response to environmental changes [13]. In the edge computing context, it signifies local data processing and analysis. Unlike traditional cloud-based intelligence, which requires data upload to remote clouds, EI enables prompt data handling, leading to significant bandwidth savings, improved privacy protection, and the ability for users to personalize applications using ML and DL models with self-generated data [14]. EI, due to its amalgamation of AI and edge computing, is emerging as a broad research area. This convergence facilitates real-time applications like visual analytics and medical imaging, as well as services prioritizing privacy, such as personal data systems. EI can execute tasks via edge servers, peer devices, or an edge-cloud cooperation paradigm, substantially reducing the need for data transmission to the cloud. EI provides a ubiquitous platform free from resource constraints [14]. This feature enables it to intelligently support applications and services, evolving beyond conventional platforms and adapting to distributed architectures. These systems allow components to collaborate, creating ubiquitous intelligence and delivering composed services to users, other devices, and applications. The potential of EI spans various domains like autonomous vehicles, industrial IoT, healthcare, and more. Integrating EI into these heterogeneous systems brings benefits including improved latency, enhanced privacy and security, reduced dependency on network connectivity, the potential for offline operation, and an array of localized, personalized smart services to enhance user experiences.

2.2 Manifestations of Edge Intelligence

EI revolves around four core aspects: *edge caching*, *edge training*, *edge inference*, and *edge offloading* [11]. Each aspect has a distinct role, but they interact seamlessly to augment the application of intelligence at the network's edge.

Edge caching is the EI component responsible for data collection and storage. It functions as a distributed data system located near end users, processing and analyzing data from various edge devices and the internet to support intelligent applications. The diverse and heterogeneous nature of these data sources calls for efficient strategies for caching—what to cache, where to cache, and how to cache.

Training in ML involves developing models or algorithms based on available data. In the context of EI, this process transforms into edge training. Here, models or algorithms are trained either on a single device or collaboratively across multiple edge devices. The heterogeneity of devices is a key consideration in edge training, with optimization modules addressing challenges related to update frequency, cost, and privacy/security issues.

Edge inference, a crucial aspect of this chapter, refers to the application of trained models or algorithms to infer outputs from given inputs. This process typically involves performing a "forward pass" through the model to compute output values based on input data. However, in the EI paradigm, edge inference must adapt to the constraints of less powerful edge devices. Therefore, achieving efficient edge inference often involves model adaptations, such as compression, and techniques (e.g., model pruning, model quantization, knowledge distillation [15]) to accelerate real-time responses, recognizing the diverse and heterogeneous nature of edge environments. This process of training and inference is visually summarized in Fig. 2. In our exploration, we focus primarily on AI inference tasks that are fully executed within a single node. This approach differs from techniques where the AI model's inference operation is split and distributed across multiple nodes [16].

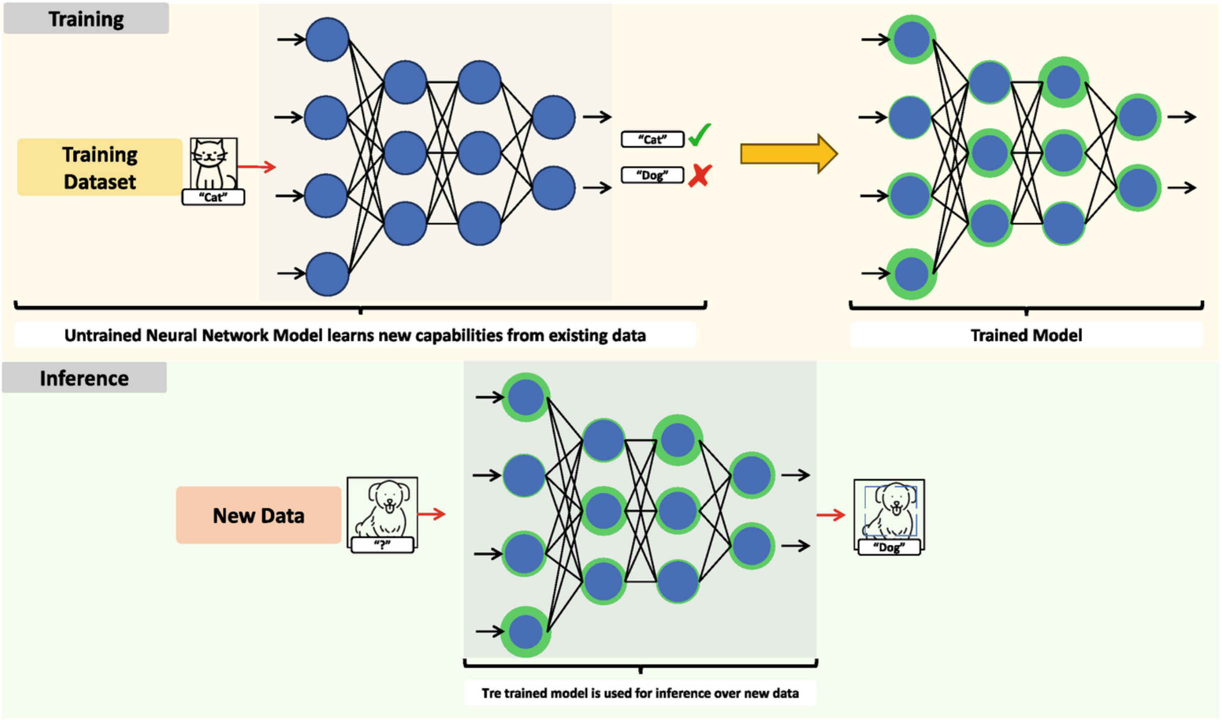

Training
Training Dataset
"Cat"
"Cat"
"Dog"
Untrained Neural Network Model learns new capabilities from existing data
Trained Model
Inference
New Data
"?"
"Dog"
Tre trained model is used for inference over new data

Fig. 2 Comparative illustration of AI training and inference: This diagram delineates the distinctive stages of ML training, wherein models are developed based on existing data, and ML inference, where these trained models are utilized to derive insights from new input data

Finally, edge offloading is a distributed computing paradigm crucial for EI. It involves allocating tasks from resource-limited edge devices to more capable edge servers or other devices. This aspect is particularly important considering the heterogeneity of edge environments, as it optimizes the utilization of available resources. Edge offloading integrates various computing architectures, including device-to-edge (D2E), device-to-device (D2D), and edge-to-edge (E2E).

In essence, these four aspects of EI represent its ability to adapt to and thrive within diverse and complex edge environments. Their interplay forms the basis for the intelligent transformation of traditional computational paradigms at the network's edge.

2.3 From Edge Intelligence to Edge AI

Having explored the manifestations and applications of EI, it is vital to delve into a closely related, yet distinct term that is often intertwined with it: Edge AI. It is essential, however, to clarify the difference between the terms "Edge AI" and "Edge Intelligence," as both are widely used in the context of edge computing but often interchangeably, potentially leading to confusion.

"Edge AI" focuses on the technological aspect—it signifies the execution of AI algorithms on edge computing environments, or in close proximity to the data sources. This term primarily involves the hardware and software components that enable AI tasks to be performed at the edge, such as AI models, accelerators, and related infrastructures.

On the other hand, "Edge Intelligence" embodies a more extensive concept—it signifies the intelligence encapsulated in the processing and decision-making capabilities at the edge of the network. Rather than just the hardware and software, it accounts for the ability of an edge system to process data, infer meaningful insights, and autonomously make decisions based on AI models.

It is important to note that this distinction, while purposeful in this chapter, is not universally rigorous. In the literature, various terminologies reflect the intersection of AI and edge computing, with notations such as "AI on Edge," "AI for Edge," and "Intelligent Edge" appearing with different nuanced meanings as defined in [17, 18]. We introduce this differentiation to provide a more structured understanding in the context of our discussion, enabling readers to anchor their comprehension amidst the diverse terminologies.

As we proceed through this chapter, it is important to recognize these subtleties. While discussing technical elements like AI models or accelerators, we will employ the term "Edge AI." Conversely, when delving into matters involving decision-making, autonomy, or system-level considerations, we will use the term "Edge Intelligence." By maintaining this distinction, we aim to enhance clarity and foster a deeper understanding of the intricate dynamics at play in the realm of edge computing.

Now that we have a clear understanding of EI and how Edge AI fits into its wider spectrum, it is necessary to delve deeper into the requirements and challenges that come with implementing heterogeneous Edge AI Inference—a topic we will explore in the following sections.

---

3 Heterogeneous Edge AI Inference: Benefits, Requirements, and Challenges

Building upon our understanding of EI and Edge AI, we now transition to the unique requirements and challenges posed by distributed Edge AI inference in heterogeneous systems. As introduced in the previous sections, the adoption and deployment of Edge AI systems have grown exponentially lately. Yet, achieving efficient and scalable AI inference at the edge, especially in heterogeneous and distributed environments, poses a unique set of technical challenges and requirements.

In this section, we delve into the technical and practical aspects of performing AI inference in heterogeneous and distributed edge environments. The diversity of edge devices, their varying hardware and software capabilities, and the nuances of mobile network architecture add layers of complexity to the optimization and execution of AI inference tasks. Consequently, it necessitates a deeper understanding of the requirements to ensure the efficient orchestration of services, QoS compliance, and robust security measures, all while dealing with the constraints of computational power and network latency. This exploration begins with an evaluation of the potential benefits and challenges associated with heterogeneous Edge AI inference. We then proceed to delve into the intricacies of hardware and software heterogeneity in the context of Edge AI. Further, we discuss the specific requirements and unique challenges involved in the implementation of heterogeneous Edge AI inference. Lastly, we examine some practical scenarios that highlight the importance and application of heterogeneous Edge AI inference in real-world settings.

3.1 Potential and Benefits of Heterogeneous Edge AI Inference

Diving deeper into the realm of Edge AI, the role of distributed Edge AI inference in heterogeneous systems emerges as a key focus. This approach offers a series of benefits that could redefine the landscape of AI applications in heterogeneous environments.

Primarily, distributed Edge AI inference introduces the power of decentralization, but with an added layer of complexity. In heterogeneous systems, different nodes have different types of hardware, each with its own strengths. Distributing the AI inference load across these diverse devices at the network edge allows us to circumvent the limitations of central processing and utilize the unique capabilities of each hardware type. For instance, GPUs might be leveraged for heavy parallel processing tasks (e.g., parallelized vector multiplication), CPUs can manage sequential tasks or parallelized scalar multiplication, TPUs can take on parallelized matrix multiplication, and VPUs for running machine vision algorithms. This diverse deployment enables handling a wider variety of tasks, increasing the system's versatility [19].

By bringing this aspect a step further, we can envision increasing flexibility in deploying AI applications composed of a pipeline of single-model tasks or multi-model applications. In such scenarios, the execution of these pipelines could be distributed opportunistically, with each component assigned to the most suitable part of our network. This approach opens the possibility of AI service chaining or composition/decomposition, leveraging the heterogeneous environment to its fullest potential. This ties in with the concept of load balancing. Efficient distribution of AI inference tasks based on the computational resources and task specifics of each node ensures that no single node is overwhelmed. This not only optimizes resource usage but also leads to swift response times and enhanced system performance—qualities highly sought after in applications requiring real-time or near real-time actions.

Another significant benefit of this approach lies in model sharing between nodes. By relying on an infrastructure that makes these heterogeneous Edge AI devices interoperable, it becomes feasible to share AI models across devices equipped with different AI accelerators, increasing service availability and redundancy. This presupposes the existence of tools that can adapt a model shared by one device to another with a different underlying platform, creating a more dynamic and flexible Edge AI environment.

Lastly, we should not overlook the potential for improved energy efficiency. By intelligently scheduling the distribution of inference tasks, we can better manage power utilization across devices. This is not just about balancing the load; it is also about optimizing the "power consumption per inference task" ratio, which contributes to more sustainable and cost-effective operations.

However, tapping into these advantages comes with its own set of challenges. In the following sections, we will explore the technical hurdles and practical issues involved in implementing distributed Edge AI inference in a heterogeneous environment. Moreover, we will probe potential solutions to navigate these complexities, and highlight practical scenarios that demonstrate the value of heterogeneous Edge AI inference in real-world settings.

3.2 Hardware and Software Heterogeneity in Edge AI Inference

The explosion of Edge AI chips in consumer devices, led by high-end smartphones, underscores the tremendous diversity and distributed nature of edge devices executing AI inference tasks. Intense research and development activity around these devices highlights their growing importance. A specialized class of System on a Chip (SoC), known as edge accelerators, has been tailored to efficiently run ML models on edge devices. They offer low inference latency and different optimizations, making them an integral part of advanced Edge AI systems for real-world applications. Various companies have proposed hardware solutions to accelerate machine learning algorithms at the network edge [20]. In this section, we will inspect three types of edge accelerators, designed for edge computing environments.

It is also worth mentioning that, in this context, software frameworks play a pivotal role. Numerous frameworks, such as TensorFlow [21], PyTorch [22], etc., have been developed to assist in the creation, training, and deployment of AI/ML models. These tools provide comprehensive libraries for defining and computing complex mathematical operations, particularly those needed for neural network ML models. However, in this work, we focus on TensorFlow, one of the most widely used open-source libraries for ML and neural networks.

Google's Edge TPU Google introduced its edge version of the Tensor Processing Unit (TPU) platform, known as Edge TPU [23], under the brand name Coral in 2018. This dedicated integrated circuit can deliver up to 4 Tera OPerationS (TOPS) per second while consuming only 2 watts (2 TOPS/watt). It supports only signed integer operations at 8 and 16 bits and is equipped with roughly 8 MB on-chip RAM for caching the model's parameters. However, due to its design for optimal inference, it exclusively supports TensorFlow Lite models that fulfill specific criteria [4] (e.g., parameter quantization).

NVIDIA's Jetson Nano GPU Launched in 2019, Jetson Nano [24] is a GPU-empowered board hosting a 64-bit quad-core Arm Cortex-A57 CPU and a NVIDIA Maxwell GPU with 128 CUDA-cores capable of delivering up to 472 GFLOPs of float operations. Sharing a common bank of 4 GB of LPDDR4 RAM between the CPU and GPU necessitates memory reservation tuning. This board runs a variant of the Ubuntu operating system and natively supports TensorFlow, compiled with GPU support, and TensorRT.

Intel's Movidius Myriad X VPU Myriad X VPU [25] by Intel houses a Neural Compute Engine—a dedicated hardware accelerator for DL inferences. This low-power SoC is specifically designed to accelerate DL deployments and computer vision applications. The chip features several processors and computing units optimized for high parallelism and DNN inference, allowing it to run up to 4 TOPS with a power budget of 1.5 watts.

Recognizing the unique constraints and requirements of these edge accelerators, specialized optimizations are required to fully utilize their hardware acceleration capabilities. We provide a simplified compilation workflow for these edge accelerators in Fig. 3, beginning with a TensorFlow frozen inference graph.

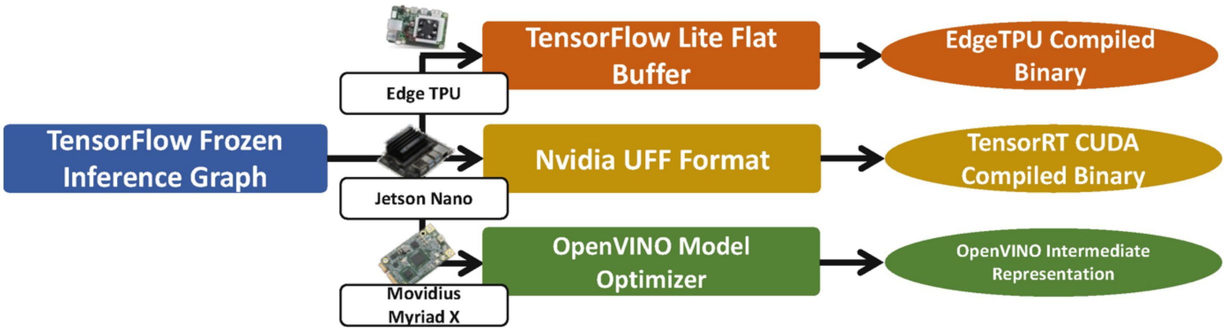

TensorFlow Frozen Inference Graph
Edge TPU
Jetson Nano
Movidius Myriad X
TensorFlow Lite Flat Buffer
Nvidia UFF Format
OpenVINO Model Optimizer
EdgeTPU Compiled Binary
TensorRT CUDA Compiled Binary
OpenVINO Intermediate Representation

Fig. 3 Simplified compilation workflow for edge accelerators: This figure illustrates the steps involved in the model conversion process for various edge accelerators, starting from a TensorFlow frozen inference graph. The specific conversion methods for Edge TPU, NVIDIA Jetson Nano (GPU), and Movidius Myriad X (VPU) are highlighted

A TensorFlow frozen inference graph is essentially a serialized version of the computation graph, which encapsulates the neural network model architecture and its trained parameters. In this state, no further training can occur, but the graph is optimized for inference instead. Freezing the graph also allows redundant nodes to be removed, enhancing execution efficiency, which is particularly beneficial for edge devices with limited computational resources. Given the TensorFlow frozen inference graph as the starting point, the conversion process for the models compatible with these devices proceeds as follows (a more thorough overview of the model conversion process can also be found in [20]):

Conversion for Edge TPU (Coral Dev Board) The model conversion for the Coral Dev Board is a two-step process. First, the TensorFlow model is transformed into a TensorFlow Lite model [26]—a lightweight and efficient variant of TensorFlow designed for hardware with constraints. Subsequently, the model undergoes Edge TPU compilation using Google's Edge TPU compiler [27]. Quantization is a crucial element in this process [28].

Conversion for Jetson Nano GPU Similar to the Edge TPU conversion, the Jetson Nano conversion is also a two-step process. The model is initially converted to Nvidia's Universal File Format (UFF) [29], with specific operations in the graph substituted with their TensorRT [30] plugin counterparts. Then, the graph is compiled into a CUDA engine, generating a binary that facilitates TensorRT-optimized inference for a particular model.

Conversion for Movidius Myriad X VPU OpenVINO [31], an open-source toolkit created by Intel, streamlines the optimization and deployment of DL models, making them compatible with Intel inference engines such as the VPU. The OpenVINO-compatible model conversion is relatively simpler than others. The sole step in the process involves converting the model to OpenVINO's Intermediate Representation format using Intel's model optimizer [32]. Importantly, this model format is not a hardware-specific compiled binary and can be executed on any hardware device compatible with any OpenVINO environment.

3.3 Requirements and Challenges for Heterogeneous Edge AI Inference

The deployment of heterogeneous Edge AI systems capable of distributed inference processing brings forth a unique set of complex requirements and challenges. While some of these are common in edge computing environments, additional, more specific challenges arise when considering the heterogeneous nature of Edge AI systems. Understanding both the common and specific challenges is crucial to designing and managing systems that are both efficient and effective.

First, hardware and software compatibility and interoperability emerge as prerequisites. The array of AI accelerators involved, such as TPUs, VPUs, and GPUs, demands robust interoperability among all components. Similarly, software—spanning AI/ML models to operating systems—needs to function seamlessly across all devices. However, achieving this harmonization is complicated given the wide variety of device manufacturers, and software providers and the involvement of different generations of technology.

The necessity to identify and provide AI inference services in a platform-agnostic manner also poses a significant challenge. Regardless of the AI accelerator that eventually processes it, an end device seeking Edge AI inference should effectively locate the required algorithm within the edge cluster. The heterogeneity of devices can compound the complexity of service discovery and provisioning.

Next, the diverse software toolsets and APIs, which are utilized to tailor, optimize, and interact with the models, add layers of complexity to the system. In response, a unified API or management framework could streamline these interactions. Designing such a unified system across a broad range of devices and platforms is a substantial challenge. Different platforms may have unique constraints or require specific approaches that necessitate careful consideration.

Effective resource management is a pivotal aspect of these systems. Edge devices not only have diverse computational capabilities and power efficiencies, but they can also be specifically suited for different types of tasks due to the hardware heterogeneity (e.g., GPU, VPU, TPU). This increases the system’s versatility and allows it to handle a greater variety of tasks. However, such heterogeneity necessitates the development of complex resource management and job scheduling strategies. These strategies must optimally allocate tasks to the best-suited resources while satisfying QoS requirements. This task becomes even more intricate when additional factors such as QoS requirements, power efficiency, mobile network latency, and more are considered.

In these systems, which frequently rely on real-time or near-real-time inference, low-latency communication capable of managing high bandwidth loads becomes necessary. However, ensuring such communication in a network with varying capabilities can be challenging due to network constraints and variations in device capabilities. It is critical to note that the overall QoS is influenced not only by the execution of AI inference at the edge but also by network latency. Two primary components contribute to this latency: one is the AI inference execution time, and the other is the latency incurred as end devices forward AI inference requests to the edge. Both of these factors are crucial in managing the system's performance and efficiency, and it is paramount to consider them in the design and management of heterogeneous Edge AI systems.

One of the cornerstones of distributed systems is the assurance of data security and privacy. This challenge is magnified in a heterogeneous Edge AI system due to the variety of devices, software, and data types involved. Additionally, in the face of potential hardware or software failures, systems must exhibit fault tolerance and reliability. Designing such a system in a heterogeneous environment can be complex, as failures in one device could potentially impact the entire system.

Finally, considerations of scalability and energy efficiency come to the forefront. As more devices and applications are added to the edge network, the system needs to be scalable while maintaining performance and efficiency. This is a considerable challenge, particularly as the number and diversity of devices increase. Considering many edge devices are battery-powered, the system should strive for energy efficiency. However, achieving energy efficiency in a system with varying device capabilities can be challenging, as some devices may consume more power than others, impacting the overall energy efficiency of the system.

In summary, the complexity of managing a heterogeneous Edge AI system arises from a wide range of challenges, from device compatibility and software harmonization to efficient resource management and low-latency communications. While it is beyond the scope of a single study to address all these challenges, the compelling advantages of such systems motivate us to make strides wherever possible. This paper attempts to tackle selected challenges, thereby contributing to the larger goal of harnessing the benefits of heterogeneous Edge AI systems. The next sections will delve into our approach and its potential implications for the future of Edge AI.

3.4 Practical Scenarios for Heterogeneous Edge AI Inference

To fully understand the potential and significance of distributed Edge AI inference in heterogeneous context, we can consider several practical scenarios in which AI accelerators are utilized. These scenarios range from Smart Cities filled with Autonomous Vehicles (AVs) to immersive Virtual Reality (VR) experiences [33], and even Unmanned Aerial Vehicles (UAVs) employed for surveillance operations [34].

AV in Smart Cities AVs form a critical component in the emerging landscape of Smart Cities, significantly depending on real-time data processing and AI inference for secure and efficient navigation within urban terrains. Envision a network interconnecting AVs, traffic management systems, surveillance cameras, and intelligent traffic lights, each equipped with distinct AI tasks and capabilities. During rush hour, an AV dealing with a heavy computational load due to complicated traffic situations might offload non-essential AI tasks (such as long-term route optimization) to other less engaged nodes in the network, like a nearby traffic management system. This task offloading, steered by a real-time Orchestration Engine, allows the AV to concentrate its computational resources on immediate navigation tasks, thereby guaranteeing safety and performance. The feature of service discovery becomes critical in this situation, as each AV must swiftly identify accessible services for task offloading. Furthermore, the system must exhibit fault tolerance to manage unexpected service disruptions without jeopardizing the vehicle or its occupants.

Immersive VR Experiences The realm of VR introduces a plethora of interesting applications, ranging from gaming to distance learning and training simulations. In these settings, Edge AI can significantly boost the user experience by minimizing latency and improving real-time processing. For example, a complex VR game could offload its heavy AI tasks to nearby underutilized edge devices, ensuring fluid gameplay without any lags or interruptions. Such a VR system would need to promptly discover accessible services to maintain the immersive experience. Additionally, it must be capable of tolerating failures and smoothly transitioning to alternative services, emphasizing the significance of fault tolerance in maintaining a continuous user experience.

UAV for Surveillance UAVs outfitted with advanced vision systems are frequently used for surveillance and monitoring tasks, particularly in demanding or remote locations. In situations where multiple UAVs are operating simultaneously, a central command center can function as an orchestrator, synchronizing the efforts of these UAVs. If a UAV is involved in a complex object tracking task, straining its computational resources, it might offload other tasks such as long-term trajectory planning or video encoding to the command center or another UAV with available capacity. This distributed inference-processing approach allows for efficient resource allocation and task completion. In this context, service discovery becomes crucial as the UAV needs to promptly locate available services for task offloading. Moreover, the system needs to be fault-tolerant to manage the potential failure of a UAV or a node and to ensure the uninterrupted execution of the overall operation.

In summary, these scenarios provide practical demonstrations of the capabilities of distributed Edge AI inference in heterogeneous settings. They highlight the necessity for effective service discovery and provisioning mechanisms, efficient resource management strategies, and a robust fault-tolerance design for successful deployment of distributed AI inference at the edge.

4 A Framework for Inference Provisioning in Heterogeneous Edge AI

Having established the benefits, requirements, and challenges associated with implementing distributed Edge AI inference in heterogeneous environments, we now move forward to present a proposed solution to help address some of these complexities. It's pertinent to note that solutions like OpenEI have started addressing similar challenges. OpenEI, as introduced by [35], is a lightweight software platform aiming to equip edges with intelligent processing, data sharing capabilities, inference, and optimized model selection. One of the strengths of OpenEI is its “libei” library, which provides a uniform RESTful API, ensuring that the architecture's heterogeneity remains transparent to the user. This allows for simplified data sharing and collaboration between edge devices. However, while OpenEI has made significant strides, it still lacks certain functionalities.

Specifically, it does not support the seamless and real-time migration of AI inference services across heterogeneous devices, AI inference service discovery, edge devices resource management, or inter-edge node orchestration.

With this backdrop, this section outlines our framework designed to facilitate the provisioning of AI inference tasks in such diverse edge settings. First, we introduce the concept of service networking, explicating its critical relevance and the key features that make it pivotal in the context of AI inference provisioning. Subsequently, we present our proposed framework, designed to tackle the multifaceted nature of Edge AI. The objective is to provide a comprehensive approach for the discovery, provisioning, and orchestration of AI inference services across a diverse array of distributed edge devices. We emphasize the framework's vital role in managing heterogeneous edge nodes effectively, each with unique hardware capabilities, software requirements, and operating under varied conditions.

Shifting from the theory to practice, we illustrate the applicability of our framework through a series of case studies. These real-world scenarios underline the capability of our framework in different application scenarios. We start by exploring how our framework leverages DNS-based edge inference discovery for seamless service identification and connection. We also delve into how it ensures system health and robustness for reliable inference provisioning. Then, we illustrate how our solution orchestrates single-model and multi-model Edge AI inferences, effectively managing the complexities of heterogeneous Edge AI and intricate AI workflows.

To conclude this section, we revisit the requirements and challenges previously discussed and elaborate on how our proposed framework addresses them, outlining the potential for future enhancements.

4.1 Service Networking for Edge AI Inference Provisioning

To address the challenges of Inference Provisioning in Heterogeneous Edge AI environments, service networking tools can play a crucial role. These tools serve as the backbone infrastructure for orchestrating seamless interaction amongst different services within a distributed system. By leveraging service networking tools, Edge AI systems can maintain QoS-compliant performance while tackling the complexity of the system. A key aspect of service networking in this context is service discovery. In a heterogeneous Edge AI environment, devices with different capabilities might be present, each potentially running different AI accelerators such as GPU, VPU, and TPU. Therefore, an efficient service discovery mechanism is needed to identify which device can provide the needed service, i.e., run a specific AI inference task, in a timely and efficient manner. In our solution, for instance, we utilize a tool like Consul for dynamic service registration and discovery [36].

Another key feature enabled by service networking is load balancing. With different nodes having varied processing capabilities and loads within a heterogeneous Edge AI system, service networking tools can support the intelligent distribution of tasks across these nodes, ensuring optimal resource utilization and sustained system performance. The next feature is health checking. Service networking also empowers constant health monitoring of various nodes in the system, identifying and mitigating potential failures before they escalate into significant issues. This is particularly critical in Edge AI environments where consistent performance and high availability are paramount. Furthermore, service networking allows for seamless integration and interoperability among different devices. As each edge device might have its own set of hardware and software requirements, a tool like Consul can provide a unified interface for interaction, thereby reducing the complexity of system operation. Furthermore, its platform’s cross-compatibility and ease of installation, which cover a wide range of environments from 386 to AMD64, ARM, and ARM64, make it particularly suitable for heterogeneous Edge AI systems. Finally, service networking tools secure intra-system communications. In the realm of Edge AI, where sensitive data might be transmitted between devices, this capability is crucial. Tools like Consul offer features like service segmentation and automatic TLS encryption to ensure secure and reliable communication.

In the subsequent section, we will discuss how we leverage these capabilities of service networking in the architecture of our proposed framework for inference provisioning in heterogeneous Edge AI environments.

4.2 Framework Architecture

The architectural design of our framework encompasses three primary components and several supplementary modules (Fig. 4). This structure intends to forge an abstraction layer facilitating interoperability among diverse AI-enabled edge devices, the provision of AI inference services in a platform-agnostic fashion, and the support of seamless service orchestration and execution migration.

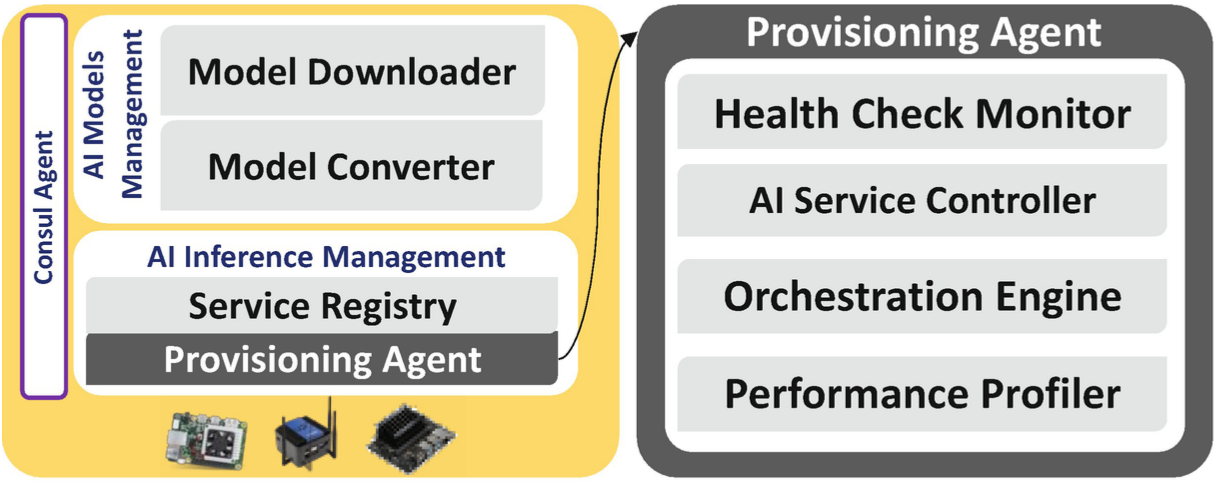

Consul Agent
AI Models Management
Model Downloader
Model Converter
AI Inference Management
Service Registry
Provisioning Agent
Provisioning Agent
Health Check Monitor
AI Service Controller
Orchestration Engine
Performance Profiler

Fig. 4 Architectural overview of the proposed framework for AI inference provisioning in heterogeneous edge environments

Consul Agent Serving as the backbone component, the Consul is used a multifaceted service networking tool that secures network interactivity among services housed across various runtimes. The Consul agent, a core process executed on every cluster node and end device, facilitates a robust service mesh with control and data planes. The control plane centralizes the service registry, regulates traffic, and coordinates system-wide operational aspects. The data plane, being distributed, handles data transmission between services per their design [37].

AI Models Management This segment deals with automating the process of downloading and converting AI models on individual edge nodes. It comprises the AI Model Downloader, developed with principles akin to the OpenVINO Model Downloader, and the AI Model Converter, designed to bridge disparities by automating several model conversion steps. The implementation of the Model Converter can be accessed at [38]. This conversion process, which is fully detailed for Google's Coral Dev Board (Edge TPU), NVIDIA Jetson Nano (TensorRT), and Intel's UP Squared AI Edge (OpenVINO), is integral to the optimization of each device's hardware.

AI Inference Management This component relies not only on Consul's capabilities but also integrates bespoke implementations developed for this study. It includes the Services Registry and the Provisioning Agent, which in turn comprise several sub-components. The Services Registry maintains a list of available AI inference services on each edge node. With Consul, services can be specified in a configuration file or added at runtime via dedicated HTTP APIs. In our configuration, AI inference services are exposed to the Service Registry via the "Provisioning Agent." This agent, built upon a foundational micro web framework, offers a RESTful API to manage lifecycle tasks of AI inference services, encompassing their activation and deactivation. By engaging with the Service Registry, the agent operates as a conduit between user requests and services. Broadly, the AI Inference Management facilitates service discovery, updates service information in the Service Registry, and assures proper command execution through the Provisioning Agent. It also optimizes operation and resource utilization locally and across different Edge AI nodes, thereby presenting a unified and efficient approach to AI inference management. Under the Provisioning Agent, we also have:

AI Service Controller Serves as the primary interface for various AI accelerators (e.g., TPU, GPU, and VPU) based on the underlying hardware-software platform. As an abstraction layer, it manages communication with different AI hardware and oversees the entire lifecycle of an inference process, tailored to the specifics of the underlying hardware/software platform. This includes preparing AI models and data for inference, initiating the inference process on suitable hardware, gathering and processing results if needed, handling hardware-specific issues, and deallocating resources post-inference.

Performance Profiler Continuously monitors and analyzes the performance of the AI inference services and the system, identifying potential bottlenecks and inefficiencies.

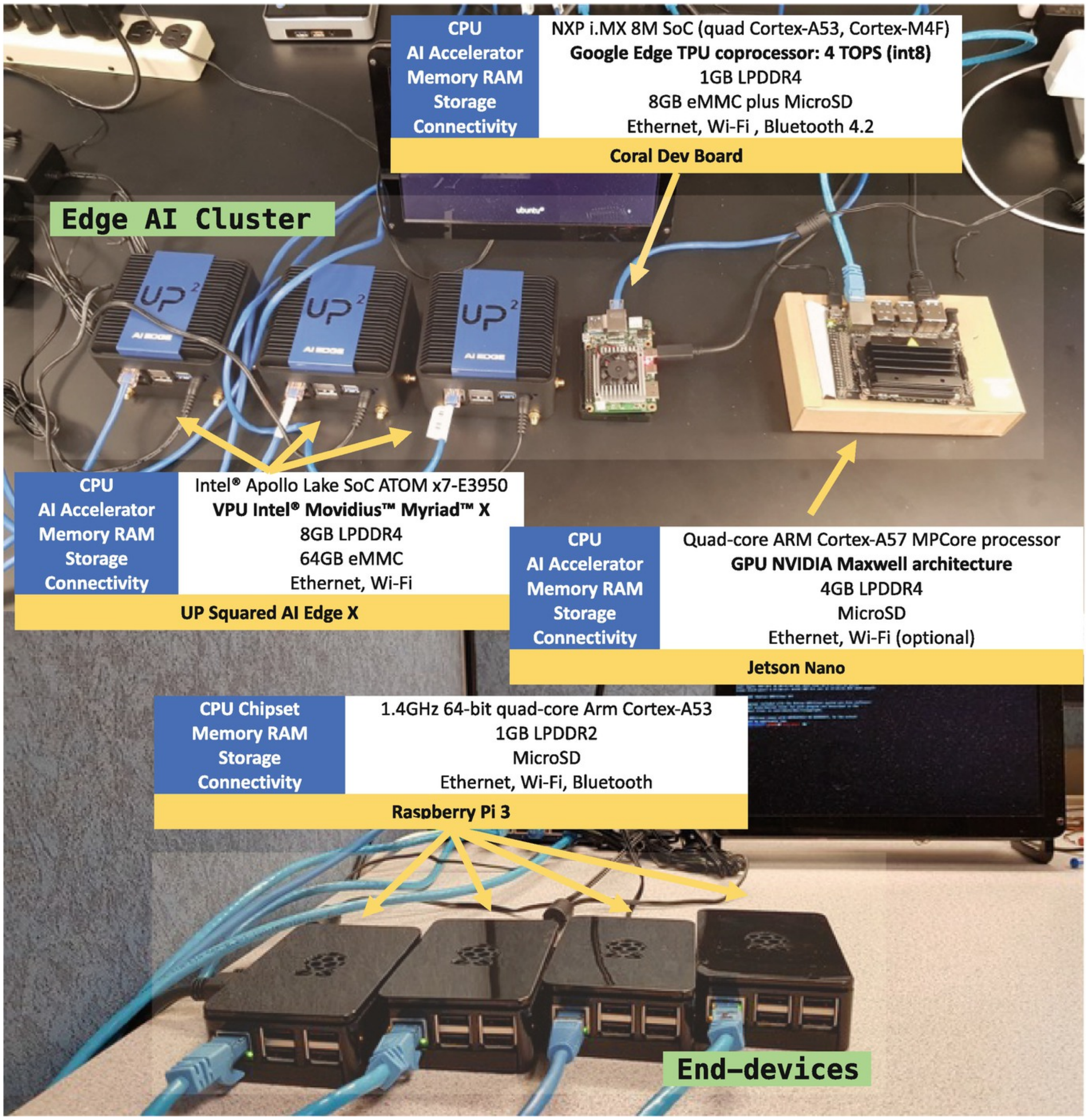
Health Check Monitor Regularly checks the health of the running AI inference services and the overall platform, alerting relevant parts of the system when potential issues are detected.

Orchestration Engine Manages and coordinates complex tasks and workflows within the system, automating processes, provisioning resources, and balancing loads.

Our proposed framework, with its architecture, aims to streamline the process of executing AI inference tasks in a heterogeneous edge environment, providing a comprehensive solution to the unique challenges associated with this task. In the following sections, we will delve into case studies that demonstrate the framework's practical application.

4.3 Test Bed and Experimental Setup

In our experimental setup, we introduce heterogeneity into our Edge AI platform via the use of three distinct AI accelerators, as previously mentioned in Sect. 3.3. These consist of the Intel Movidius Myriad X VPU, Google Edge TPU, and the NVIDIA 128-core Maxwell GPU. These cutting-edge system-on-chip devices find their homes in three separate Single-Board Computers (SBCs), namely the UP Squared AI Edge [39], Coral Dev Board [40], and the NVIDIA Jetson Nano [41]. The specific hardware arrangement of our edge computing test bed and a summary of the hardware attributes of each device are shown in Fig. 5.



Coral Dev Board	
CPU	NXP i.MX 8M SoC (quad Cortex-A53, Cortex-M4F)
AI Accelerator	**Google Edge TPU coprocessor: 4 TOPS (int8)**
Memory RAM	1GB LPDDR4
Storage	8GB eMMC plus MicroSD
Connectivity	Ethernet, Wi-Fi , Bluetooth 4.2

UP Squared AI Edge X	
CPU	Intel® Apollo Lake SoC ATOM x7-E3950
AI Accelerator	**VPU Intel® Movidius™ Myriad™ X**
Memory RAM	8GB LPDDR4
Storage	64GB eMMC
Connectivity	Ethernet, Wi-Fi

Jetson Nano	
CPU	Quad-core ARM Cortex-A57 MPCore processor
AI Accelerator	**GPU NVIDIA Maxwell architecture**
Memory RAM	4GB LPDDR4
Storage	MicroSD
Connectivity	Ethernet, Wi-Fi (optional)

Raspberry Pi 3	
CPU Chipset	1.4GHz 64-bit quad-core Arm Cortex-A53
Memory RAM	1GB LPDDR2
Storage	MicroSD
Connectivity	Ethernet, Wi-Fi, Bluetooth

Fig. 5 Edge computing test bed. The test bed showcases the deployment of five nodes including three different SBCs: the UP Squared AI Edge, Coral Dev Board, and the NVIDIA Jetson Nano. The setup is complemented by four Raspberry Pi 3 Model B acting as end devices

In addition to AI capabilities, the SBCs also exhibit heterogeneity in CPU, RAM, and storage. For our experiments, we use four Raspberry Pi 3 Model B units as end devices. These compact yet powerful devices are equipped with a 1.4 GHz 64-bit quad-core Arm Cortex-A53 CPU, 1GB of LPDDR2 RAM, and feature gigabit Ethernet, integrated 802.11ac/n wireless LAN, and Bluetooth 4.2 connectivity.

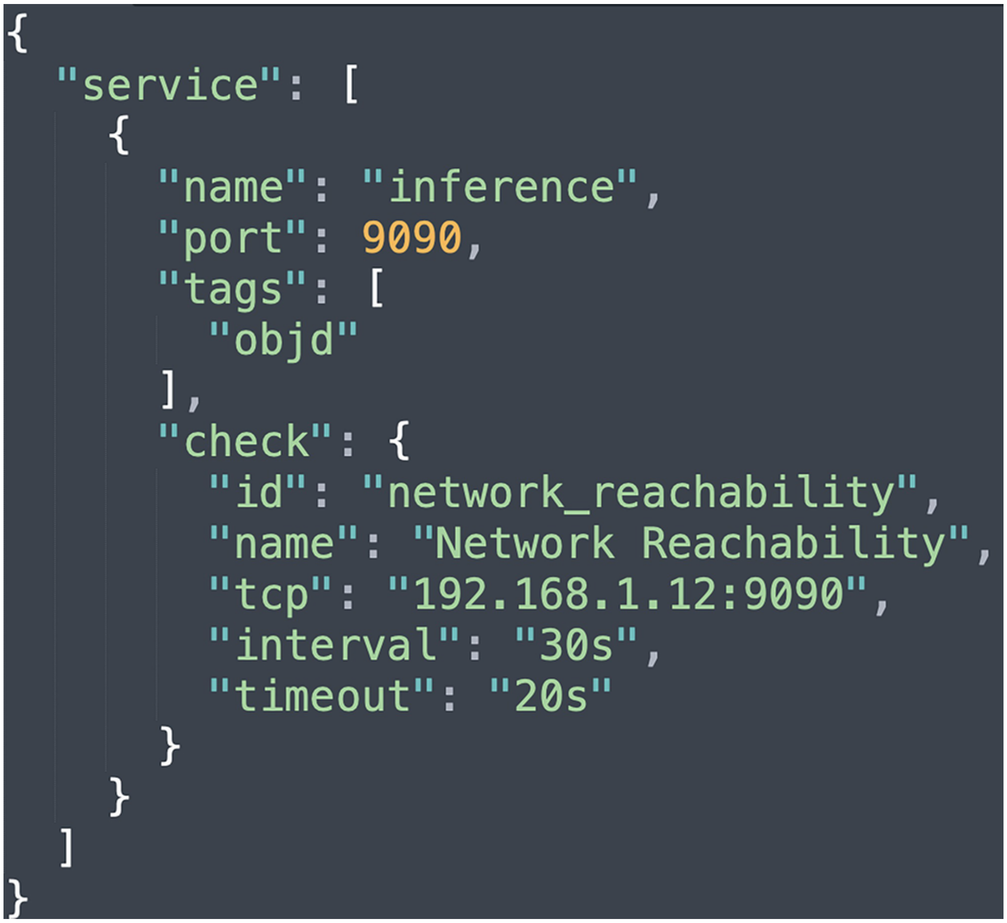
Our experimental devices are networked via a controlled wireless environment for communication. With our Edge AI test bed devices co-located, we introduce an emulation layer to mimic a realistic Edge system deployment and simulate network latency typically encountered between edge devices, as performed in [42].

In our experiments, particularly those outlined in Sect. 4.6, the AI inference application used by the edge devices incorporates the MobileNetV1 algorithm. This is applied to video content of 1200 × 1200 resolution streamed from each end device. The input to the model is a blob consisting of a single image with dimensions of (1, 300, 300, 3), where each figure in the tuple denotes batch size, image width, image height, and color channels, respectively. In the experiment in Sect. 4.7, the models used for face detection, age recognition, head-pose estimation, emotion detection, and facial landmark detection are selected from the extensive collection of pretrained models available within the OpenVINO model zoo [43].

4.4 DNS-Based Edge AI Inference Discovery

In this section, we focus on the process of discovering AI inference services within a cluster of edge devices. Within our system, this task is performed by Consul via DNS (Domain Name System) inquiries.

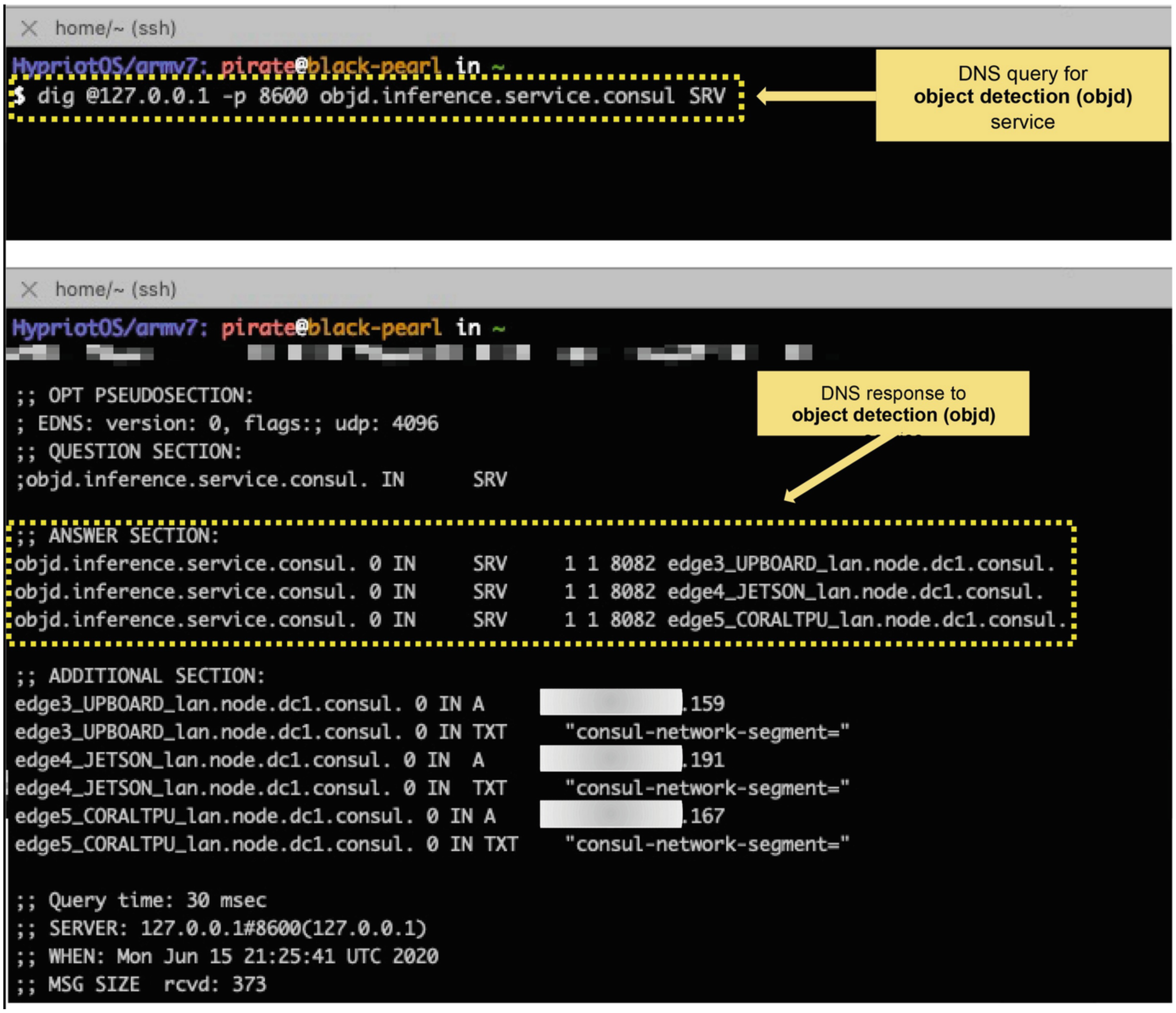
Service discovery hinges on a centralized service registry—a "directory" listing available services, their locations, and health statuses. Each service instance is accompanied by a lightweight service discovery client, enabling them to register and deregister with the central registry programmatically [44]. The registry monitors service health statuses consistently, aiding in effective availability management. Consul, in our setup, maintains an automated, up-to-date catalog of all nodes and services, including their health statuses. For service discovery implementation, service instances must register with the central registry upon deployment. If a service needs to interact with another, it consults the registry, which then provides the network locations of all available instances of the desired service. In our scenario, DNS executes this process. For service discovery to be effective, services must be defined or registered to Consul. Our network's services are defined in a configuration file, which includes a set of parameters that guide various service aspects, including discovery by other network services. Figure 6 provides an example of a service definition for an inference service.



```
{
  "service": [
    {
      "name": "inference",
      "port": 9090,
      "tags": [
        "objd"
      ],
      "check": {
        "id": "network_reachability",
        "name": "Network Reachability",
        "tcp": "192.168.1.12:9090",
        "interval": "30s",
        "timeout": "20s"
      }
    }
  ]
}
```

Fig. 6 Consul service registration JSON: Definition of an “inference” service associated with “objd” (object detection) and “fcd” (face detection) tags, with a TCP health check for network reachability

With this understanding, let’s consider an example from our scenario. Suppose an end-device requires an object detection service, “objd.” The end-device queries the DNS server directly with “objd.inference.service.consul.” This DNS inquiry searches for nodes providing the “objd” service, initiating a DNS-Based Service Discovery. The system then begins the discovery process by consulting the Service Registry, identifying operational nodes that offer the “objd” service. For instance, if a cluster with ten nodes offering the “objd” service has two nodes that fail health checks, the DNS query responds with a list of the eight operational nodes. Figure 7 illustrates a terminal output when an end-device executes a DNS query, demonstrating the simplicity and effectiveness of DNS-based service discovery in identifying available and operational ML inference services within an edge device cluster.



```
HypriotOS/armv7: pirate@black-pearl in ~
$ dig @127.0.0.1 -p 8600 objd.inference.service.consul SRV
```

```
HypriotOS/armv7: pirate@black-pearl in ~
[redacted]

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;objd.inference.service.consul. IN      SRV

;; ANSWER SECTION:
objd.inference.service.consul. 0 IN     SRV     1 1 8082 edge3_UPBOARD_lan.node.dc1.consul.
objd.inference.service.consul. 0 IN     SRV     1 1 8082 edge4_JETSON_lan.node.dc1.consul.
objd.inference.service.consul. 0 IN     SRV     1 1 8082 edge5_CORALTPU_lan.node.dc1.consul.

;; ADDITIONAL SECTION:
edge3_UPBOARD_lan.node.dc1.consul. 0 IN A       [redacted].159
edge3_UPBOARD_lan.node.dc1.consul. 0 IN TXT     "consul-network-segment="
edge4_JETSON_lan.node.dc1.consul. 0 IN  A       [redacted].191
edge4_JETSON_lan.node.dc1.consul. 0 IN  TXT     "consul-network-segment="
edge5_CORALTPU_lan.node.dc1.consul. 0 IN A      [redacted].167
edge5_CORALTPU_lan.node.dc1.consul. 0 IN TXT    "consul-network-segment="

;; Query time: 30 msec
;; SERVER: 127.0.0.1#8600(127.0.0.1)
;; WHEN: Mon Jun 15 21:25:41 UTC 2020
;; MSG SIZE  rcvd: 373
```

Fig. 7 Terminal output: DNS query execution by an end-device seeking an object detection (objd) inference service. The DNS response reveals that only three out of five nodes in the edge cluster responded, suggesting the remaining two are either not healthy or do not host the requested service

4.5 System Health Assurance for Reliable AI Inference Provisioning

Ensuring reliable inference provisioning necessitates a robust system health assurance mechanism. This mechanism, consisting of health checks and system monitoring functionalities, is designed to maintain the operation of AI inference services and the overall platform.

In this respect, understanding Consul Health Checks is essential. Health checks are predefined configurations that assess the health status of a service or node. As mentioned in [45], different types of checks can be formulated, including script checks that call an external application to conduct the health check and return an appropriate exit code, and TCP checks that attempt to connect to an IP, hostname, and port via TCP, pausing for a specific duration.

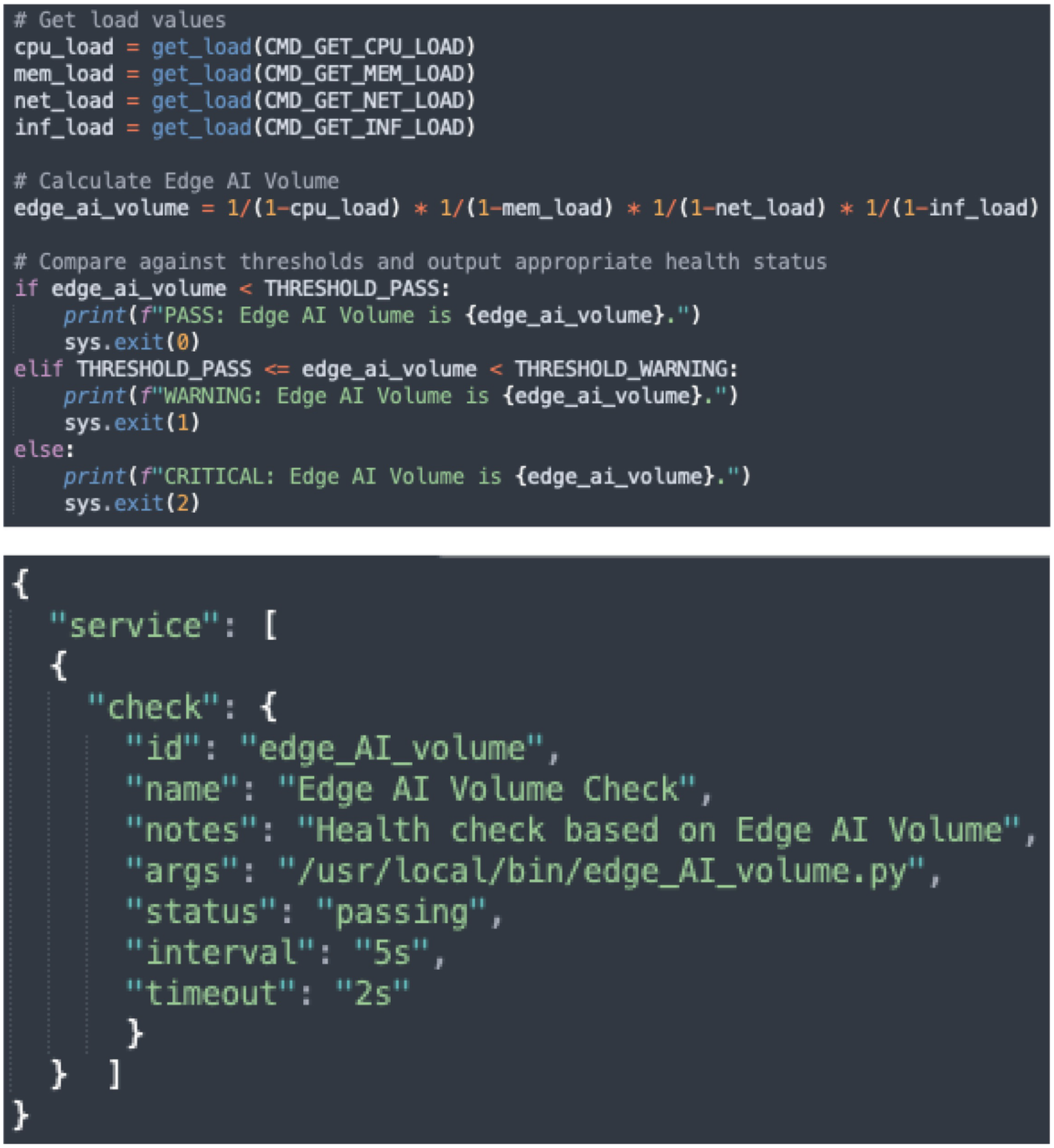
Our system's operational integrity relies heavily on the Health Check Monitor, which continuously evaluates the health and QoS compliance of the running AI inference services and the overall platform. It ensures all components and services operate as expected, verifies system and service availability, and oversees resource utilization. When potential issues arise, the Health Check Monitor alerts relevant system components to initiate reactive actions. In accordance with the Consul methodology, the health check can assign the system or an application to one of three potential levels: *pass*, *warning*, or *critical*. A critical system health status renders the device unreachable, unqueryable by other devices, or unable to participate in offloading tasks. Similarly, a *critical* inference service suggests the device is overloaded, unable to meet AI inference latency requirements.

We have developed a unique approach for setting health check thresholds in our deployment. Specifically, we introduce a new metric, the *Edge AI Volume*, representing the device's overall multidimensional load. The Edge AI Volume is computed using the Eq. (1):

$$\mathrm{Edge\ AI\ Volume} = 1/(1-\mathrm{cpu})^{*}\ 1/(1-\mathrm{mem})^{*}\ 1/(1-\mathrm{net})^{*}\ 1/\,(1-\mathrm{infC}) \tag{1}$$

This metric extends the volume concept defined in [46], accounting for the real-time AI accelerator load (infC). By monitoring volume values' impact on AI inference performance, we could set the pass, warning, and critical thresholds for each device's health checks. Similarly, the Performance Profiler can estimate the Edge AI Volume portion generated by each running inference application.

As an illustration of health check registration and definition in our system, see Fig. 8.



```
# Get load values
cpu_load = get_load(CMD_GET_CPU_LOAD)
mem_load = get_load(CMD_GET_MEM_LOAD)
net_load = get_load(CMD_GET_NET_LOAD)
inf_load = get_load(CMD_GET_INF_LOAD)

# Calculate Edge AI Volume
edge_ai_volume = 1/(1-cpu_load) * 1/(1-mem_load) * 1/(1-net_load) * 1/(1-inf_load)

# Compare against thresholds and output appropriate health status
if edge_ai_volume < THRESHOLD_PASS:
    print(f"PASS: Edge AI Volume is {edge_ai_volume}.")
    sys.exit(0)
elif THRESHOLD_PASS <= edge_ai_volume < THRESHOLD_WARNING:
    print(f"WARNING: Edge AI Volume is {edge_ai_volume}.")
    sys.exit(1)
else:
    print(f"CRITICAL: Edge AI Volume is {edge_ai_volume}.")
    sys.exit(2)
```

```
{
  "service": [
  {
    "check": {
      "id": "edge_AI_volume",
      "name": "Edge AI Volume Check",
      "notes": "Health check based on Edge AI Volume",
      "args": "/usr/local/bin/edge_AI_volume.py",
      "status": "passing",
      "interval": "5s",
      "timeout": "2s"
      }
  }  ]
}
```

Fig. 8 Core logic of the Edge AI volume-based health check and corresponding consul health check registration (full script details provided in the text)

For the health check, we first define a “get_load” function to retrieve the current system load values using appropriate system commands. We then use these values to calculate the Edge AI Volume as per Eq. (1). This calculated volume is compared against pre-set thresholds to ascertain the system’s health status (Fig. 8 displays the script’s core part). We utilize Python’s built-in sys module to exit the script with a status code corresponding to the determined health status. Customizing the “get_load” function, system load retrieval commands, and threshold values is necessary to cater to specific environments and needs. A demonstration video of this functionality is available online [47], showcasing how end-device interactions with edge devices can be affected by potential edge device overload and the subsequent role of health checks.

We employ Edge AI Volume-based health checks and a greedy-heuristic approach in our implementation, providing insight into the device's overall multidimensional load and that created by each running application. We also define network-aware health checks to ensure that end devices are served only by edge nodes with favorable network latency, thereby ensuring QoS compliance for request/response AI services.

Through this mechanism, we ensure that an overloaded edge node due to concurrent processing tasks, or one that fails to meet QoS requirements, becomes unreachable or transfers the execution of an inference instance to a healthier node. This strategy effectively maintains system health and ensures the reliable provisioning of inference.

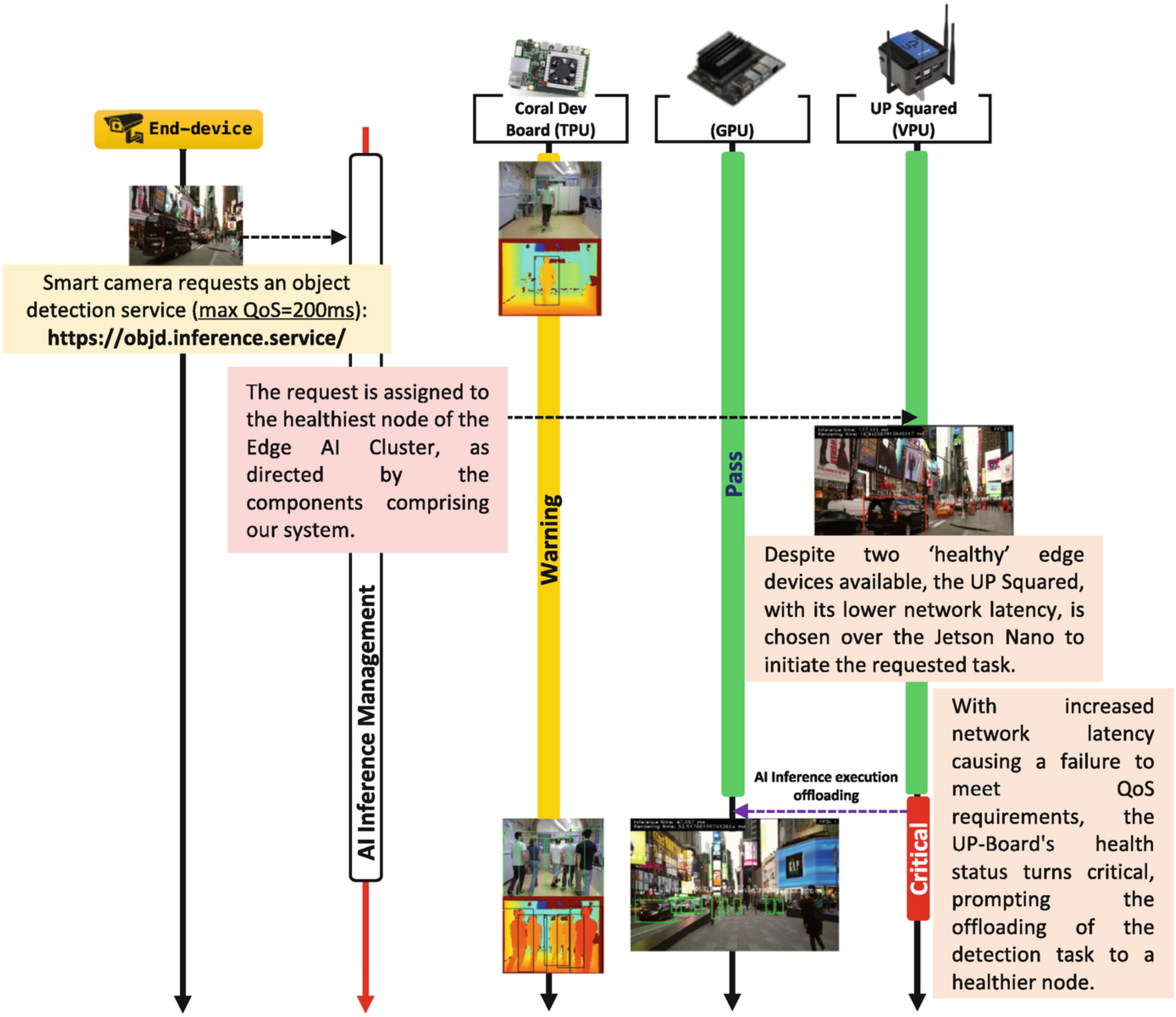
4.6 Orchestration of Single-Model Edge AI Inference

In this section, we discuss the orchestration system's capabilities for the single-model Edge AI inference case. We highlight the system's capacity to smoothly offload AI inference service execution across various edge nodes in near real-time, showcasing the system's dynamic adaptability. This migration process relies on the Orchestration Engine, which is designed to manage these transitions based on each edge node's available resources and health status, and service availability.

Our system is designed to consider other edge nodes' available computational resources within the cluster. It also factors in their respective network latency conditions related to the end device that initiated the inference request, allowing the system to reallocate inference execution to the most capable node within the cluster.

We present a scenario to illustrate this. Here, an edge node efficiently offloads the execution of an object detection algorithm—applied to a video content streamed from an end device—to a more resourceful edge node, as soon as local resource overload is detected. This smooth transition between nodes is facilitated by sharing specific metadata, which includes details about the ML model in use and the video stream source.

The underlying mechanics of the orchestration process can be better understood by examining the accompanying video [47] and the included flowchart (Fig. 9), which detail the decision-making process of the Orchestration Engine. We present two case studies showcasing our system's efficiency in adapting to varying loads and changing network conditions. In the first scenario (as shown in the video), the system's adaptability is tested under increasing loads, while the second scenario highlights the system's resilience in the face of fluctuating network conditions (Fig. 9). In the video, viewers can observe how the system smartly migrates the execution of AI inference from one edge node to another when the original device experiences resource overload or a spike in network latency.



Smart camera requests an object detection service (max QoS=200ms): **https://objd.inference.service/**

The request is assigned to the healthiest node of the Edge AI Cluster, as directed by the components comprising our system.

Despite two 'healthy' edge devices available, the UP Squared, with its lower network latency, is chosen over the Jetson Nano to initiate the requested task.

With increased network latency causing a failure to meet QoS requirements, the UP-Board's health status turns critical, prompting the offloading of the detection task to a healthier node.

Fig. 9 Flowchart detailing the decision-making process of the Orchestration Engine, illustrating the system's adaptability under varying network conditions

QoS requirements play a vital role in this orchestration process. For this demonstration, we have chosen an arbitrary QoS value of 200ms. However, it is important to remark that QoS constraints can be adjusted to be more or less stringent, depending on specific case requirements.

The *Edge AI Volume*, by offering an estimation of the load attributable to each running application, aids in identifying services that excessively demand resources, thereby facilitating their migration to maintain system balance and performance.

In addition to the demonstration, we provide empirical data representing the average inference latency for each edge device when handling multiple instances of the object detection application simultaneously (Fig. 10). As the number of instances increases, the demand for edge device resources escalates, often resulting in a violation of the set QoS requirements. The influence of hardware heterogeneity on performance also becomes evident. This underlines the necessity of our system's orchestration capabilities, which manage service migration to healthier nodes, ensuring adherence to QoS requirements despite the increased load.

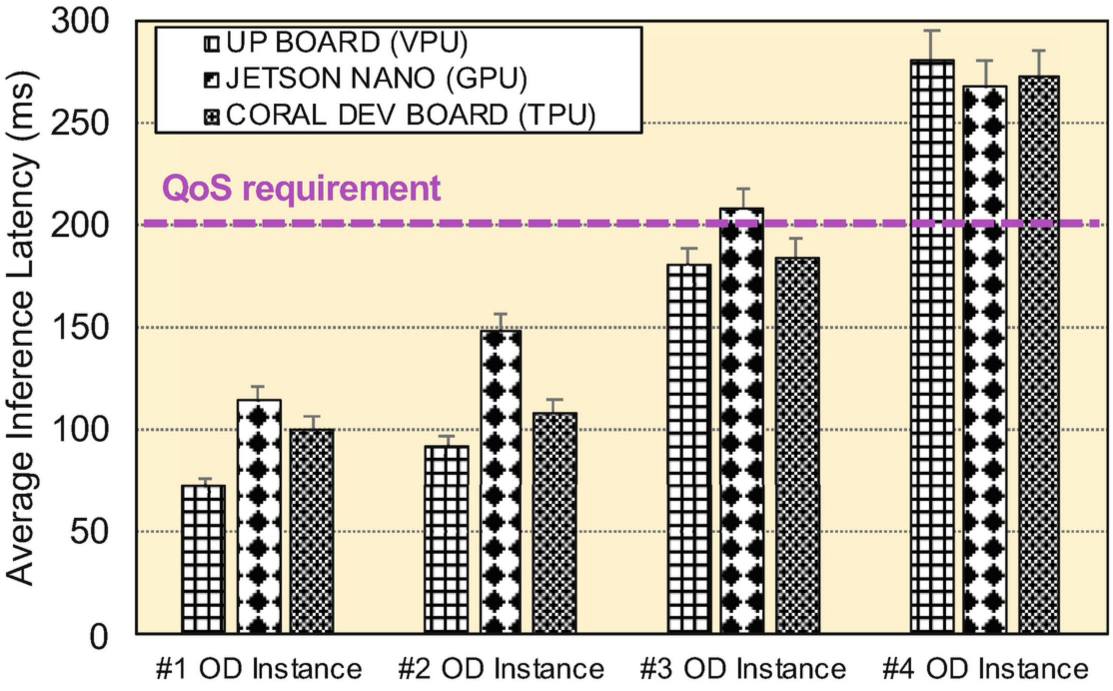


Fig. 10 Graph illustrating the average inference latency for each edge device while handling multiple instances of the object detection (OD) application. This figure highlights the influence of hardware heterogeneity and the impact of concurrent processing tasks on meeting set QoS requirements

In summary, our system exhibits impressive capabilities in orchestrating real-time offloading of AI inference service execution across a variety of edge nodes. When processing video content streamed from an end-device, the system can smoothly transfer the execution of an object detection algorithm to a more resource-rich edge node at the first signs of local overload. This migration, which considers both the computational resources and network latency conditions of edge nodes relative to the requesting end-device, enhances the system's adaptability and overall performance. This solidifies its position as a reliable tool for managing single-model Edge AI Inference.

4.7 Orchestration of Multi-model Edge AI Inference Pipelines

In this section, we delve into our system's proficiency in managing multi-model Edge AI Inference Pipelines, underscoring the additional capabilities of the Orchestration Engine. Specifically, we present a unique scenario where our system adeptly partitions the execution of a complex inference pipeline across distinct edge devices.

One important aspect to remark is that each pipeline is built around a primary machine learning model, which forms the foundation for the entire pipeline. Subsequent, compatible models are integrated into this base structure, enhancing its functionality. To illustrate, our demonstration pipeline incorporates five distinct models. The cornerstone model of this pipeline is a face detection algorithm, supplemented by age recognition, head-pose estimation, emotion detection, and facial landmarks detection to create an exhaustive facial analytics pipeline. Notably, these supplementary models are designed to work seamlessly with the base face detection result.

In our demonstration scenario, we employ two UP Squared AI Edge boards to showcase how our system distributes the execution of a pipeline among disparate edge devices. This intelligent partitioning is pivotal in averting an overload on any single edge device, thus ensuring compliance with the stipulated QoS requirements.

To further elucidate this concept, we provide an informative video [47] and an explanatory flowchart (Fig. 11). The video serves as a practical demonstration of how our system orchestrates multi-model AI inference pipelines, effectively balancing the computational load across various edge devices. This functionality is particularly useful when handling intricate inference tasks that necessitate simultaneous execution of multiple machine learning models, as demonstrated in our facial analytics pipeline. The accompanying flowchart offers a comprehensive visual depiction of the decision-making process underlying the Orchestration Engine. Together with the video, this flowchart clarifies the process of managing multi-model Edge AI inference pipelines, offering transparent insight into the engine's capabilities.

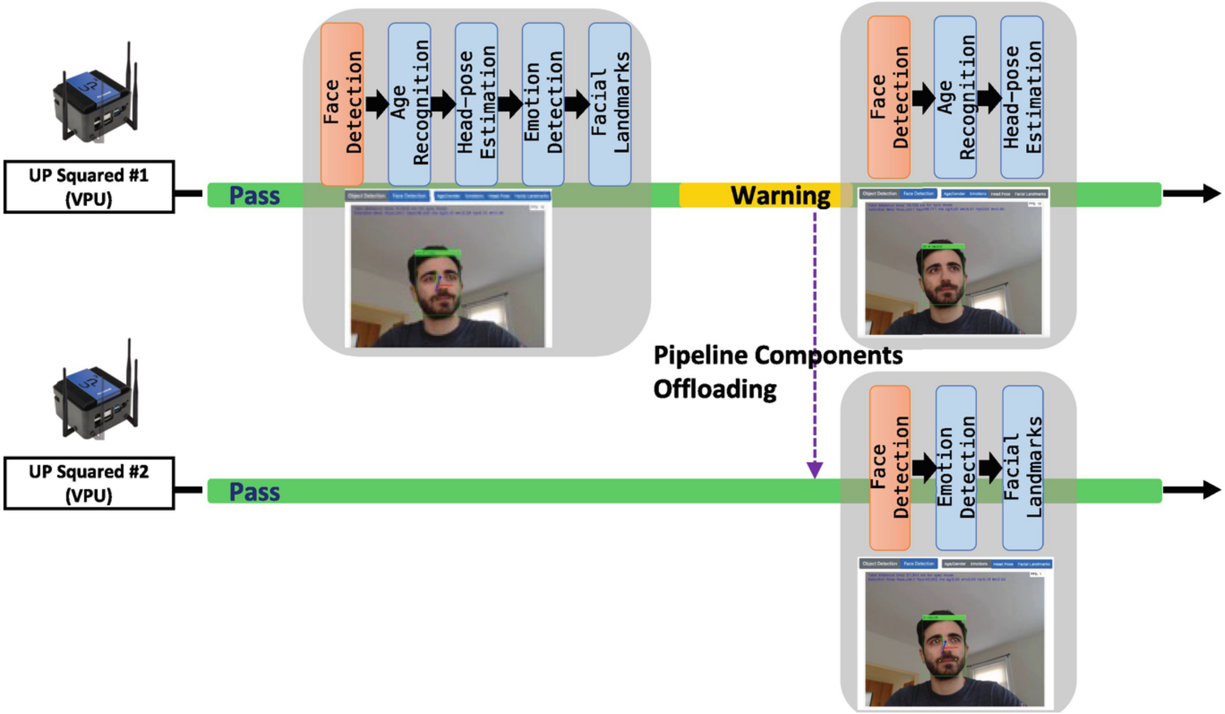

Face Detection
Age Recognition
Head-pose Estimation
Emotion Detection
Facial Landmarks
UP Squared #1 (VPU)
Pass
Warning
Face Detection
Age Recognition
Head-pose Estimation
Pipeline Components Offloading
UP Squared #2 (VPU)
Pass
Face Detection
Emotion Detection
Facial Landmarks

Fig. 11 The UP Squared #1 node operates a multi-model Edge AI inference pipeline comprising five models (baseline model plus four additional components). As computational demand escalates, threatening to breach QoS parameters, the health status of UP Board #1 shifts to "warning." This triggers a partial offload of the pipeline to a healthier node (UP Squared #2), restoring the health status of UP Squared #1

A significant detail in this process is the critical role played by the Orchestration Engine. This engine determines when and where to relocate specific components of the pipeline based on the computational load each component imposes and the overall multidimensional load of the edge devices (quantified by the *Edge AI Volume*). The engine ensures that the moved components of the pipeline shift the status of the original node from "critical" to at least "warning" or "pass." Additionally, the relocated pipeline components are moved to a node that is in a "healthy" state. This smart orchestration aids in preventing local overload, preserving the system's adaptability, and upholding overall performance, even under significant computational demands.

The outcomes from this distributed inference process, executed across two nodes, are then gathered through a publish/subscribe (pub/sub) mechanism. This method ensures efficient data aggregation and accurate pipeline output.

In summary, our system showcases the capability of managing the execution of intricate, multi-model inference pipelines across multiple edge devices. This capacity is crucial for improving the system's adaptability, maintaining QoS requirements under substantial computational load, and reinforcing its efficacy in orchestrating multi-model Edge AI inference.

4.8 Addressing Requirements and Challenges

As we draw this section to a close, it is worthwhile to revisit the requirements and challenges outlined in Sect. 3. The solution we have proposed confronts several of these effectively, while others offer avenues for future exploration and development.

Our system aims to mitigate interoperability issues through a unified management framework and API, facilitating interactions with a diverse range of AI accelerators through a streamlined and cohesive interface. The system manages the discovery and provisioning of AI inference services in a hardware-agnostic fashion, thereby catering effectively to the diverse range of hardware present in a heterogeneous environment. Additionally, the presence of an Orchestration Engine empowers our system to manage resources and job scheduling effectively. The engine accounts for the computational capabilities of each edge device and service availability, all while ensuring adherence to QoS requirements.

In response to the demand for real-time or near-real-time Edge AI inference, our solution incorporates a simple heuristic approach that capably manages varying network conditions under typical circumstances. However, more complex situations involving significant network latency might require alternative and more robust solutions.

Fault tolerance and reliability have also been woven into our solution's design. However, as the system's distribution expands, maintaining these aspects becomes increasingly challenging and presents a fruitful area for further research and development. This observation extends to scalability as well. Although our system is designed to scale and handle increased data volume, sustaining performance and efficiency remains a significant hurdle as we contend with increasing numbers and diversity of devices and applications.

In conclusion, our proposed solution aptly addresses several key requirements for distributed Edge AI Inference and successfully tackles certain challenges. Nevertheless, the lingering challenges underline the necessity for and opportunities for ongoing research and development in the field.

---

5 Conclusion and Future Directions

As we reflect on the chapter’s content, the significance of streamlining the provisioning of AI inference services in heterogeneous Edge AI systems becomes clear, essential for expanding and refining a diverse range of applications in EI. Our proposed system, while effectively tackling several challenges, highlights areas that demand more in-depth exploration.

Challenges tied to network latency, energy efficiency, scalability, and fault tolerance provide ample opportunities for future development in this domain. Recent years have seen both industry and academia making significant strides toward establishing an edge-to-cloud continuum through the technological enhancement of orchestration tools. However, the open challenges we have identified underscore the necessity of supplementing existing orchestration and service management frameworks with additional features. These features should be capable of addressing the escalating need for the provisioning of varied AI inference services within heterogeneous computing and networking environments. Furthermore, considering the emergent trends in AI and ML, like the advent of TinyML [48], which integrates AI inference capabilities into embedded devices, there is an impending need to broaden orchestration capabilities. Such future designs should contemplate extending AI-specific orchestration to these “beyond edge” devices [49].

The influence of heterogeneity on performance, especially across different inference applications, is another significant aspect to consider. This would necessitate a reevaluation of how orchestration rules are defined, thereby adding a new dimension to our understanding and approach. Future developments should aim to devise mechanisms that favor automated scaling configurations based on the performance and accuracy requirements of inference queries. These mechanisms, while fulfilling application needs, must also optimize resource management in SBC clusters that often operate under energy constraints. Notable efforts, like the utilization of INFaaS for generating model-variants [50], highlight the potential in this direction. This approach allows for efficient navigation of the large trade-off space of AI model-variants to meet application-specific objectives. However, these solutions are primarily suitable for high-resource computing environments. Although it would be possible to adapt its design guidelines for constrained edge scenarios, there are additional hurdles that must be overcome, such as limited storage and multi-tenant capabilities.

Regarding our heuristic strategy pivoted around health checks, it is evident that enhancements are feasible. Future optimizations could also factor in the ability of edge devices to handle various types of applications, complex network conditions, and the use of combined *quantitative metrics* (e.g., AI inference latency) and *qualitative metrics* (e.g., ML algorithms accuracy) [51]. Further challenges could include the integration of real-time data analysis and decision-making into these systems, balancing the need for local processing power with the potential benefits of cloud-based AI resources, and ensuring robust security and privacy protections in increasingly distributed and diverse networks.

In conclusion, while our solutions provide a robust foundation, the domain is ripe for ongoing research and development, offering promising prospects for advancements in Edge AI.

References

1. Zhou, Z., et al.: Edge intelligence: paving the last mile of artificial intelligence with Edge computing. Proc. IEEE. **107**(8), 1738–1762 (2019)
[Crossref]
2. Chen, J., et al.: Deep learning with Edge computing: a review. Proc. IEEE. **107**(8), 1655–1674 (2019)
[Crossref]
3. NVIDIA: What's the Difference Between Deep Learning Training and Inference? https://blogs.nvidia.com/blog/2016/08/22/difference-deep-learning-training-inference-ai/ (2016)
4. Chen, X., et al.: ThriftyEdge: resource-efficient Edge computing for intelligent IoT applications. IEEE Netw. **32**(1), 61–65 (2018)
[Crossref]

5. ITU-T, 2020: 2030-Gap Analysis of Network 2030 New Services, Capabilities and Use Cases

6. Sipola, T., Alatalo, J., Kokkonen, T., Rantonen, M.: Artificial intelligence in the IoT era: a review of Edge AI hardware and software. In: Proceedings of the 31st Conference of Open Innovations Association, pp. 320–331. IEEE (2022)

7. Deloitte Insights; Bringing AI to the Device: Edge AI Chips Come into Their Own. https://www2.deloitte.com/us/en/insights/industry/technology/technology-media-and-telecom-predictions/2020/ai-chips.html

8. Morabito, R., Chiang, M.: Discover, provision, and orchestration of machine learning inference services in heterogeneous edge. In: Proceedings of the IEEE 41st International Conference on Distributed Computing Systems, pp. 1116–1119. IEEE (2021)

9. Barbuto, V., Savaglio, C., Chen, M., Fortino, G.: Disclosing Edge intelligence: a systematic meta-survey. Big Data Cognit. Comput. **7**(1), 44 (2023)
[Crossref]

10. Motlagh, N.H., Lovén, L., Cao, J., Liu, X., Nurmi, P., Dustdar, S., Tarkoma, S., Su, X.: Edge computing: the computing infrastructure for the smart megacities of the future. Computer. **55**(12), 54–64 (2022)
[Crossref]

11. Xu, D., Li, T., Li, Y., Su, X., Tarkoma, S., Jiang, T., Crowcroft, J., Hui, P.: Edge intelligence: empowering intelligence to the edge of network. Proc. IEEE. **109**(11), 1778–1837 (2021)
[Crossref]

12. Shi, W., Dustdar, S.: The promise of edge computing. Computer. **49**(5), 78–81 (2016)
[Crossref]

13. Legg, S., Hutter, M.: A collection of definitions of intelligence. Front. Artif. Intell. Appl. **157**, 17 (2007)

14. Peltonen, E., Ahmad, I., Aral, A., Capobianco, M., Ding, A.Y., Gil-Castineira, F., Gilman, E., Harjula, E., Jurmu, M., Karvonen, T., Kelanti, M.: The many faces of Edge intelligence. IEEE Access. **10**, 104769–104782 (2022)
[Crossref]

15. Sharma, D., Sarkar, S.: In: Pal, S., De, D., Buyya, R. (eds.) Artificial Intelligence-Based Internet of Things Systems, vol. 1, pp. 267–283. Springer, Cham (2022)
[Crossref]

16. He, Q., Dong, Z., Chen, F., Deng, S., Liang, W., Yang, Y.: Pyramid: enabling hierarchical neural networks with edge computing. In: Proceedings of the ACM Web Conference 2022, pp. 1860–1870 (2022)
[Crossref]

17. Deng, S., Zhao, H., Fang, W., Yin, J., Dustdar, S., Zomaya, A.Y.: Edge intelligence: the confluence of edge computing and artificial intelligence. IEEE Internet Things J. **7**(8), 7457–7469 (2020)
[Crossref]

18. Wang, X., Han, Y., Leung, V.C., Niyato, D., Yan, X., Chen, X.: Convergence of edge computing and deep learning: a comprehensive survey. IEEE Commun Surv Tutor. **22**(2), 869–904 (2020) [Crossref]

19. Chen, T., Moreau, T., Jiang, Z., Zheng, L., Yan, E., Shen, H., Cowan, M., Wang, L., Hu, Y., Ceze, L., Guestrin, C.: TVM: an automated end-to-end optimizing compiler for deep learning. In: *Proceedings of the 13th USENIX Symposium on Operating Systems Design and Implementation*, pp. 578–594. OSDI 18 (2018)

20. Antonini, M., Vu, T.H., Min, C., Montanari, A., Mathur, A., Kawsar, F.: Resource characterisation of personal-scale sensing models on Edge accelerators. In: Proceedings of the First International Workshop on Challenges in Artificial Intelligence and Machine Learning for Internet of Things, pp. 49–55 (2019) [Crossref]

21. TensorFlow. https://www.tensorflow.org/

22. PyTorch. https://pytorch.org/

23. Google Cloud. Edge TPU – Run Inference at the Edge. https://cloud.google.com/edge-tpu

24. Jetson Nano Developer Kit. https://developer.nvidia.com/embedded/jetson-nano-developer-kit

25. Intel. Intel® Movidius™ Myriad™ X Vision Processing Unit (VPU). https://www.intel.com/content/www/us/en/products/details/processors/movidius-vpu/movidius-myriad-x/products.html

26. TensorFlow Lite. https://www.tensorflow.org/lite/guide

27. Edge TPU Compiler. https://coral.ai/docs/edgetpu/compiler/

28. TensorFlow. Post-training quantization. https://www.tensorflow.org/lite/performance/post_training_quantization

29. NVIDIA. UFF Converter. https://docs.nvidia.com/deeplearning/tensorrt/api/python_api/uff/uff.html

30. NVIDIA. TensorRT. https://developer.nvidia.com/tensorrt-getting-started

31. OpenVINO. https://docs.openvino.ai/2023.0/home.html

32. OpenVINO. Convert a Model. https://docs.openvino.ai/2023.0/openvino_docs_MO_DG_ Deep_Learning_Model_Optimizer_DevGuide.html

33. Lin, Y., Zhang, Z., Tang, H., Wang, H., Han, S.: Pointacc: efficient point cloud accelerator. In: *Proceedings of the MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 449–461 (2021) [Crossref]

34. Bandelier, K., et al.: Use Cases for Vehicle-to-Vehicle (V2V) Communications for Unmanned Aircraft Systems, pp. 1–24 (2023)

35. Zhang, X., Wang, Y., Lu, S., Liu, L., Shi, W.: OpenEI: An open framework for edge intelligence. In: *Proceedings of the 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 1840–1851. IEEE (2019)

36. Consul. https://www.consul.io/

37. HashiCorp White Paper. Service mesh and microservices networking. https://www.hashicorp.com/resources/service-mesh-microservices-networking

38. Edge Model Converter. https://github.com/melhabr/edge-model-converter

39. UP Squared AI Vision X. https://up-shop.org/up-squared-re-ai-vision-x-developer-kit-8-gb.html

40. Coral Dev Board. https://coral.ai/products/dev-board/

41. NVIDIA. Jetson Nano Developer Kit for AI and Robotics. https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-nano/

42. Nguyen, H.T., Morabito, R., Kim, K.T., Chiang, M.: On-the-fly resource-aware model aggregation for federated learning in heterogeneous edge. In: Proceedings of the 2021 IEEE Global Communications Conference, pp. 1–6. IEEE (2021)

43. OpenVINO Model Zoo. https://docs.openvino.ai/2023.0/model_zoo.html

44. HashiCorp White Paper. Modern Service Networking for Cloud and Microservices. https://www.hashicorp.com/resources/modern-service-networking-cloud-microservices

45. HashiCorp. Consul—Define Health Checks. https://developer.hashicorp.com/consul/docs/services/usage/checks

46. Wood, T., Shenoy, P., Venkataramani, A., Yousif, M.: Sandpiper: black-box and gray-box resource management for virtual machines. Comput. Netw. **53**(17), 2923–2938 (2009) [Crossref]

47. YouTube. Discover, Provision, and Orchestration of Machine Learning Inference Services in Heterogeneous Edge. https://www.youtube.com/watch?v=xYntGeLmCSI

48. Warden, P., Situnayake, D.: Tinyml: Machine Learning with Tensorflow Lite on Arduino and Ultra-Low-Power Microcontrollers. O’Reilly Media (2019)

49. Doyu, H., Morabito, R., Brachmann, M.: A tinymlaas ecosystem for machine learning in IOT: overview and research challenges. In: Proceedings of the 2021 International Symposium on VLSI Design, Automation and Test, pp. 1–5. IEEE (2021)

50. Romero, F., Li, Q., Yadwadkar, N.J., Kozyrakis, C.: INFaaS: automated model-less inference serving. In: Proceedings of the 2021 USENIX Annual Technical Conference, USENIX ATC 21, pp. 397–411 (2021)

51. Liu, Q., Huang, S., Opadere, J., Han, T.: An edge network orchestrator for mobile augmented reality. In: Proceedings of the IEEE INFOCOM 2018-IEEE Conference on Computer Communications, pp. 756–764. IEEE (2018)

[Crossref]

OceanofPDF.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
S. Pal et al. (eds.), *IoT Edge Intelligence*, Internet of Things: Technology, Communications and Computing
https://doi.org/10.1007/978-3-031-58388-9_3

Artificial Intelligence-Enabled Edge Computing: Necessity of Next Generation Future Computing System

Anand Kumar Mishra[1], R. Ravinder Reddy[2] ✉, Amit Kumar Tyagi[3] ✉ and Micheal Olaolu Arowolo[4]

(1) Computer Science and Engineering, NIIT University, Neemrana, Rajasthan, India
(2) Chaitanya Bharathi Institute of Technology, Hyderabad, Telengana, India
(3) Department of Fashion Technology, National Institute of Fashion Technology, New Delhi, India
(4) Department of Electrical Engineering and Computer Science, Bond Life Sciences Center, University of Missouri, Columbia, MO, USA

✉ **R. Ravinder Reddy**
Email: rravinderreddy_cse@cbit.ac.in

Keywords Artificial intelligence – Edge computing – Internet of things – Next generation computing system – Smart applications

---

1 Introduction

Artificial intelligence (AI) finds applications in diverse sectors such as healthcare, finance, transportation, customer service, and beyond. AI refers to the simulation of human intelligence in machines programmed to think and learn like humans. It encompasses a wide array of techniques and technologies that empower machines to perceive, reason, learn, and make decisions. There are various types of AI, including:

- Narrow AI: Also known as weak AI, it is specialized in executing specific tasks within a limited domain. Instances of weak AI applications include voice assistants, image recognition systems, and recommendation algorithms.
- General AI: Also known as strong AI, it aspires to attain a level of intelligence and cognitive capabilities equivalent to humans. General AI has the capacity to comprehend, learn, and apply knowledge across diverse domains, adapting to a wide range of tasks and situations.
- Machine Learning: It is a subset of AI that involves the development of algorithms that enable machines to learn from data and make predictions or decisions without explicit programming. Machine learning algorithms can identify patterns and make informed decisions.
- Deep Learning: A subset within machine learning, deep learning concentrates on artificial neural networks inspired by the human brain. These algorithms, known as deep learning algorithms, can autonomously derive representations from extensive datasets and have demonstrated important points of success in domains like image and speech recognition.

Note that AI finds application in diverse industries such as healthcare, finance, transportation, customer service, and beyond. Its transformative potential extends to various facets of our lives, enhancing efficiency, accuracy, and decision-making processes.

1.1 Edge Computing

Edge computing is a decentralized computing paradigm that involves bringing data processing and storage closer to the point of data generation. In traditional cloud computing models, data is transmitted to remote servers or data centers for processing and analysis. In contrast, edge computing conducts processing at or near the edge devices, such as sensors, IoT devices, or edge servers, located in proximity to the data source. The fundamental concept behind edge computing is to diminish latency and bandwidth requirements by locally processing data at the edge rather than shuttling it back and forth to a centralized cloud infrastructure. By handling data closer to its origin, edge computing enables real-time or near-real-time analysis and response, proving particularly advantageous for time-sensitive applications or situations with limited or unreliable network connectivity. The advantages offered by Edge computing are as follows:

- *Reduced Latency:* Through local data processing, edge computing minimizes the round-trip time for data between edge devices and the cloud. This is important for applications demanding immediate or nearly immediate processing, including scenarios like autonomous vehicles, industrial automation, or real-time monitoring systems.
- *Bandwidth Optimization:* Edge computing optimizes bandwidth utilization and diminishes data transmission costs by decreasing the volume of data transferred to the cloud. This holds particular importance in situations where network connectivity is restricted or comes at a high cost.
- *Enhanced Privacy and Security:* Edge computing facilitates local data processing, minimizing the necessity of transmitting sensitive data to the cloud. This fortifies privacy and security by maintaining data in close proximity to its source, thereby reducing the vulnerability to potential security breaches or data leaks.
- *Offline Operation:* Edge computing empowers applications to function seamlessly even in the absence of network connectivity. Local processing and storage capabilities enable devices to operate autonomously, guaranteeing uninterrupted functionality in remote or disconnected environments.

Note that edge computing is mainly relevant in industries such as manufacturing, healthcare, transportation, and IoT deployments, where low latency, real-time analytics, and data privacy are important. It complements cloud computing by distributing computational capabilities across the network and using the advantages of both centralized cloud resources and local edge devices. However, note that not all computing tasks are suitable for edge deployment. Some computationally intensive or data-intensive tasks may still be better suited for cloud-based processing due to the higher computational power and storage capacities available in the cloud infrastructure. In summary, edge computing represents an evolving paradigm that brings processing and intelligence closer to the data source, enabling faster, more responsive, and more efficient computing capabilities for various applications.

1.2 Edge Computing vs Fog vs Cloud Computing

Edge computing, fog computing, and cloud computing represent three distinct computing paradigms characterized by differences in architecture, processing location, and suitability for various types of applications. Here’s a comparison of these three models:

1.2.1 Cloud Computing

Cloud computing refers to the centralized storage and processing of data on remote servers or data centers. In the cloud model, data is sent over the network to the cloud infrastructure for processing, analysis, and storage. Cloud computing provides virtually unlimited storage capacity, high computational power, and scalability. It is well-suited for applications that require massive data storage, complex computations, and data processing that does not require real-time or low-latency response.

Advantages:

- *Scalability:* Cloud computing provides elastic resources, allowing users to scale up or down based on their needs.
- *Accessibility:* Data and applications can be accessed from anywhere with an internet connection.
- *Cost-effective:* Users can avoid upfront hardware and infrastructure costs.

Limitations:

- *Latency:* The round-trip time for data transmission between the edge devices and the cloud can introduce latency, which may not be suitable for real-time or latency-sensitive applications.
- *Bandwidth Requirements*: Cloud computing relies on transferring large amounts of data over the network, which can be costly and inefficient in scenarios with limited bandwidth or high data volumes.

1.2.2 Edge Computing

Edge computing involves processing data at or near the edge devices or sensors, closer to the data source. By performing computations locally, edge computing reduces latency and minimizes the need for transferring data to the cloud. Edge computing is ideal for applications that require real-time or near-real-time processing, low latency, and efficient bandwidth usage.

Advantages:

- *Low Latency:* Edge computing enables real-time or near-real-time processing, reducing the latency associated with cloud-based processing.

- *Bandwidth Optimization:* Through local data processing, edge computing minimizes the amount of data that must be sent to the cloud, optimizing the utilization of bandwidth.
- *Offline Operation:* Edge devices can continue to function even in scenarios with limited or no network connectivity.

Limitations:

- *Limited Computational Power:* Edge devices usually possess constrained processing power and storage capacity in comparison to cloud infrastructure, potentially limiting the complexity of computations that can be executed.
- *Scalability:* Expanding edge computing infrastructure across a multitude of devices can pose challenges owing to complexities in management and coordination.

1.2.3 Fog Computing

Fog computing extends the concept of edge computing by introducing a hierarchical architecture that incorporates intermediate computing nodes, referred to as fog nodes, situated between edge devices and the cloud. These fog nodes take on the responsibilities of processing and analysis, bringing computational capabilities closer to the edge devices while maintaining the utilization of the cloud for specific tasks. Fog computing is well-suited for applications that demand a balance between low latency, local processing, and access to cloud resources.

Advantages:

- *Low Latency:* Fog computing reduces latency by bringing processing closer to edge devices, similar to edge computing.
- *Scalability:* Fog nodes can be deployed at various levels of the network hierarchy, providing scalability and management advantages compared to edge-only deployments.
- *Resource Offloading:* Fog nodes can offload certain processing tasks to the cloud, using its higher computational power and storage capacity.

Limitations:

- *Complexity:* The hierarchical nature of fog computing introduces additional complexities in terms of management, coordination, and deployment.
- *Network Congestion:* The increased number of intermediate fog nodes can introduce network congestion and increase the complexity of data

routing and communication.

To summarize, cloud computing is well-suited for applications needing extensive storage, high computational power, and scalability. Edge computing is ideal for low-latency, real-time processing in proximity to the data source. Fog computing presents a hierarchical architecture that integrates advantages from both edge and cloud computing, striking a balance between local processing and access to cloud resources. The selection of a computing paradigm depends on the specific requirements at hand.

1.2.4 AI Based Edge Computing

AI-driven edge computing merges the capabilities of artificial intelligence (AI) algorithms with edge computing infrastructure, facilitating intelligent and real-time data processing directly at the edge devices. This approach brings AI capabilities in close proximity to the data source, reducing latency and network bandwidth demands and enabling localized decision-making and analysis. Here’s how AI and edge computing can be combined:

- *Localized AI Inference:* AI models can be embedded in edge devices to conduct local inference. Rather than transmitting data to the cloud for processing, AI algorithms can operate directly on edge devices, facilitating real-time decision-making. This proves particularly beneficial in applications like autonomous vehicles, industrial automation, or smart surveillance systems, where rapid responses and low latency are paramount.
- *Data Filtering and Pre-processing:* Leveraging edge computing involves preprocessing and filtering data before transmitting it to the cloud. Local analysis by AI algorithms enables the identification of pertinent patterns and the extraction of essential features, thereby decreasing the volume of data requiring transmission to the cloud. This optimization not only minimizes bandwidth usage but also alleviates the computational load on the cloud infrastructure.
- *Privacy and Security:* By performing AI processing at the edge, sensitive data can be kept locally, enhancing privacy and security. Instead of transmitting sensitive data to the cloud, edge devices can process and analyze data without compromising privacy or exposing sensitive information to potential security threats.

- *Dynamic Model Adaptation:* AI models deployed at the edge can adapt and learn from the local data in real-time. Edge devices can continuously update their AI models based on the changing environment or user-specific requirements. This enables personalized and context-aware AI capabilities without the need for constant communication with the cloud.
- *Collaborative Edge Networks:* Multiple edge devices within a network can collaborate and share information to improve overall AI performance. By using the collective intelligence of edge devices, distributed AI models can be trained and deployed to optimize system-wide performance and provide more accurate and robust AI predictions.

Hence, AI-driven edge computing offers numerous benefits, including diminished latency, heightened privacy, optimized bandwidth utilization, and improved real-time decision-making capabilities. It empowers intelligent and autonomous behavior at the edge by harnessing the capabilities of AI algorithms. Nevertheless, it is important to acknowledge the challenges associated with AI-based edge computing, such as constraints in computational resources, complexities in management, and the imperative for efficient model deployment and updates across a distributed network. Addressing these challenges requires careful consideration of the hardware capabilities, network infrastructure, and AI model optimization techniques tailored for edge deployments. In summary, AI-based edge computing holds great potential to enable intelligent and responsive applications in various domains, extending the capabilities of AI beyond centralized cloud architectures and unlocking new possibilities for edge-enabled AI systems.

1.2.5 AI Based Edge Computing for Next Generation Computing Systems

AI-driven edge computing is recognized as an important facilitator for the next generation of computing systems. By combining the strengths of artificial intelligence (AI) and the advantages of edge computing, it gives rise to intelligent and responsive systems marked by enhanced performance, efficiency, and capabilities. Here are some ways AI-based edge computing contributes to next-generation computing systems:

- *Real-Time and Low-Latency Processing:* Implementing AI algorithms at the edge facilitates real-time and low-latency processing. By positioning

AI capabilities in close proximity to the data source, edge computing diminishes the time needed for data processing and analysis. This capability supports immediate responses and proves important for time-sensitive applications in emerging technologies such as autonomous vehicles, robotics, augmented reality, and Internet of Things (IoT) devices that demand instant decision-making and real-time interaction.

- *Intelligent Decision-Making at the Edge:* AI-based edge computing empowers edge devices to make intelligent decisions independently. Instead of relying solely on cloud-based processing, edge devices can use AI algorithms to analyze data locally, identify patterns, and make context-aware decisions. This distributed decision-making capability enhances autonomy, reduces dependence on cloud connectivity, and allows for more efficient and localized operation of computing systems.
- *Efficient Bandwidth Utilization:* Edge-based AI algorithms have the capability to preprocess and filter data before sending it to the cloud. Through local execution of initial data analysis and feature extraction, edge computing minimizes the volume of data that must traverse the network. This optimization not only enhances bandwidth usage but also diminishes latency. Such a capability proves particularly advantageous for applications that generate substantial data, including video surveillance systems or sensor networks.
- *Enhanced Privacy and Security:* AI-driven edge computing tackles privacy and security issues by locally processing data. Rather than transmitting sensitive or personal data to the cloud, edge devices can execute AI inference on the device itself, maintaining control over sensitive information. This localized processing serves to mitigate the risk of data breaches, bolster privacy, and facilitate compliance with data protection regulations.
- *Federated Learning and Collaborative Intelligence:* AI-driven edge computing enables collaborative intelligence among edge devices. Leveraging federated learning techniques, edge devices can collaboratively train AI models while safeguarding data privacy. Each device contributes its own local knowledge, and the amalgamated model is deployed back to the edge devices, enhancing the overall intelligence of the system. This collaborative approach fosters distributed and adaptive learning, allowing for continuous model improvement without solely relying on central servers.

- *Resilience and Offline Operation:* AI-based edge computing systems are capable of operating offline or in disconnected environments. Edge devices can continue to function autonomously even when there is limited or no network connectivity. This resilience ensures uninterrupted operation in scenarios where cloud connectivity is unreliable, such as remote locations or during network disruptions.

Hence, AI-based edge computing is poised to revolutionize various domains, including healthcare, transportation, smart cities, industrial automation, and more. It enables intelligent and responsive computing systems that can handle real-time data processing, make localized decisions, optimize bandwidth utilization, enhance privacy, and operate autonomously. By using the combined power of AI and edge computing, next-generation computing systems will be more efficient, capable, and adaptable to diverse application requirements.

1.2.6 Features and Characteristics

AI-based edge computing for next-generation computing systems encompasses several distinct features and characteristics that contribute to its effectiveness and impact. Here are some key features and characteristics of AI-based edge computing:

- *Immediate Decision-Making and Low-Latency Processing:* AI-driven edge computing achieves real-time and low-latency processing by positioning AI capabilities in close proximity to the data source. This facilitates instant decision-making, swift response times, and enhances overall system performance, especially in time-sensitive applications.
- *Localized Intelligence:* With AI-based edge computing, intelligence is embedded directly into edge devices. This localization of intelligence enables autonomous decision-making at the edge, reducing the need for constant communication with the cloud and enabling edge devices to operate independently and make localized decisions based on local data and context.
- *Distributed Architecture:* AI-driven edge computing utilizes a distributed architecture, distributing computational tasks across multiple edge devices. This distribution of processing alleviates the load on individual devices, optimizes resource utilization, and facilitates the creation of scalable and flexible computing systems.

- *Bandwidth Optimization:* Conducting AI processing at the edge ensures that only pertinent and essential data is transmitted to the cloud. This minimizes the volume of data sent over the network, optimizing bandwidth usage and mitigating the associated costs and latency linked to data transmission.
- *Privacy and Security:* Edge computing with AI addresses privacy and security issues by locally processing data on edge devices. This method ensures that sensitive data remains within the confines of the edge device, diminishing the risk of data breaches and unauthorized access. Additionally, it facilitates compliance with privacy regulations and grants users' greater control over their data.
- *Offline Operation and Resilience:* Edge devices equipped with AI capabilities can continue to operate even in disconnected or low-connectivity environments. They can perform AI processing and decision-making autonomously, ensuring uninterrupted operation and resilience to network disruptions.
- *Collaborative Intelligence:* AI-based edge computing facilitates collaborative intelligence among edge devices. Through techniques like federated learning, edge devices can collectively train AI models while preserving data privacy. This collaboration allows for knowledge sharing, model improvement, and collective decision-making, enhancing the overall intelligence and capabilities of the computing system.
- *Adaptive and Dynamic Learning:* AI-based edge computing enables adaptive and dynamic learning at the edge. Edge devices can continuously learn and update their AI models based on changing conditions, user interactions, or new data. This flexibility allows edge devices to adapt and improve their intelligence over time, resulting in more accurate and personalized responses.
- *Resource Efficiency:* AI-driven edge computing enhances the efficiency of computing resources. Conducting AI processing at the edge diminishes the necessity for continuous data transfer to the cloud, leading to more effective resource utilization and decreased energy consumption.
- *Contextual Awareness:* AI-enabled edge devices have the capability to analyze data within the context of their immediate environment. This contextual awareness empowers edge devices to make more informed and relevant decisions, leveraging localized information and considering specific situational factors.

Hence, these features and characteristics of AI-based edge computing contribute to the advancement of next-generation computing systems, enabling real-time processing, localized intelligence, optimized resource utilization, improved privacy and security, offline operation, collaborative learning, and adaptive decision-making.

1.2.7 Benefit and Limitations

Next-generation computing systems offer several benefits and capabilities that drive innovation and advancement in various domains. However, they also come with certain limitations. Here are some benefits and limitations of next-generation computing systems:

Benefits:

- *Increased Performance:* Next-generation computing systems use advanced technologies and architectures to deliver superior performance compared to traditional systems. They often incorporate high-speed processors, parallel computing, optimized algorithms, and accelerated hardware, resulting in faster processing speeds and improved overall system performance.
- *Enhanced Efficiency:* Next-generation computing systems are designed with a focus on efficiency. They aim to optimize resource utilization, reduce power consumption, and minimize the wastage of computing resources. By using technologies like AI-based edge computing, these systems achieve better energy efficiency and operational cost savings.
- *Real-Time and Low-Latency Processing:* Next-generation computing systems enable real-time and low-latency processing, making them suitable for time-sensitive applications. They use technologies such as AI-based edge computing and parallel processing to deliver immediate responses, enabling real-time decision-making and interactive experiences.
- *Scalability and Flexibility:* Next-generation computing systems are often designed with scalability and flexibility in mind. They can efficiently handle increased workloads and accommodate the growing demand for computational resources. With cloud computing, virtualization, and containerization, these systems can scale up or down based on demand, providing flexibility and adaptability to changing requirements.
- *Advanced Analytics and AI Capabilities:* Next-generation computing systems enable advanced analytics and AI capabilities. They use machine

learning, deep learning, and data mining techniques to extract information, detect patterns, and make accurate predictions from large amount of data. This enables businesses and organizations to gain/ extract valuable information and make data-driven decisions.

Limitations:

- *Complexity:* Next-generation computing systems can be complex to design, implement, and manage. They often require specialized skills and expertise to set up and maintain, making them challenging for organizations with limited technical resources or knowledge. Managing the various components, optimizing performance, and ensuring compatibility across the system can be complex tasks.
- *Cost:* The adoption of next-generation computing systems can involve important upfront costs. The advanced hardware, software, and infrastructure required for these systems can be expensive, especially for smaller organizations or startups with limited budgets. Ongoing operational costs, such as maintenance, upgrades, and licensing fees, can also add to the overall cost.
- *Security and Privacy Issues:* Next-generation computing systems introduce new security and privacy challenges. With increased connectivity, data sharing, and integration of diverse technologies, there is a higher risk of cyber threats, data breaches, and unauthorized access. Ensuring the security and privacy of sensitive data becomes a critical issue that requires robust security measures and protocols.
- *Data Management and Compliance:* Next-generation computing systems generate and process large amount of data. Managing and organizing this data effectively can be a complex task. Additionally, complying with data protection regulations, such as GDPR or HIPAA, becomes important to safeguard privacy and ensure legal and ethical data handling practices.
- *Legacy System Integration:* Integrating next-generation computing systems with existing legacy systems can pose challenges. Legacy systems may have different architectures, technologies, or compatibility issues that hinder seamless integration with new technologies. Migrating data, ensuring interoperability, and maintaining system cohesiveness can require careful planning and execution.

Note that we need to consider these benefits and limitations when adopting and implementing next-generation computing systems. While they offer essential advantages in terms of performance, efficiency, real-time processing, scalability, and advanced capabilities, organizations must also address the complexities, costs, security issues, data management, and integration challenges associated with these systems.

---

2 Literature Review

The rise of edge computing solutions is driven by the inadequacy of traditional cloud computing to meet the diverse data processing needs of today's intelligent society. It introduces a novel paradigm for conducting computations at the network's edge, emphasizing proximity to both consumers and data sources, setting it apart from cloud computing. This approach is tailored for local, small-scale data processing and storage at the network's edge. The referenced article [1] predominantly explores research and findings related to edge computing, starting with a comparison to cloud computing. It then discusses edge computing's architecture, keyword technology, security, and privacy protection, culminating in an overview of its applications.

Cloud-based services often utilize data centers as central servers to process data from edge devices like smartphones, tablets, and wearables. However, the model presented in [2] raises issues about its impact on quality-of-service and experience due to escalating demands on computing and communication infrastructure. The untapped computational power of edge nodes such as base stations, routers, and switches inspires the concept of edge computing, aiming to distribute some of the computational load to the network's edge. This position paper explores the opportunities and challenges arising from this shift in the computing industry.

Additionally, [3] provides a comprehensive analysis of current research on mobile edge computing (MEC), focusing on integrated radio and computational resource management. The paper addresses various issues and potential paths for MEC research, including the deployment of MEC systems, cache-enabled MEC, mobility management for MEC, green MEC, and privacy-aware MEC. The paper anticipates that advancements in these areas will facilitate the transition of MEC from theory to practice.

Concluding their discussion, they explore ongoing Mobile Edge Computing (MEC) standardization initiatives and typical MEC application scenarios. The concept of edge computing, involving data processing at the network's edge, has emerged in response to the proliferation of the Internet of Things (IoT) and the widespread use of robust cloud services. Edge computing effectively addresses challenges related to quick response times, constraints on battery life, reductions in bandwidth costs, as well as issues about data security and privacy. In the referenced article [4], edge computing is defined, accompanied by diverse case examples that illustrate its practical implementation. These cases span from cloud offloading and smart cities to homes and collaborative edge scenarios.

The increasing necessity of edge computing, where data processing occurs partially at the network edge rather than solely in the cloud, is driven by the growing prevalence of the Internet of Things and advanced cloud services. This approach holds potential to alleviate challenges associated with latency, limited battery life on mobile devices, bandwidth costs, and issues about security and privacy [5].

Furthermore, the surge in industry investment and research interest in recent years attests to the heightened importance of edge computing. Positioned at the Internet’s edge near mobile devices or sensors, this technology, as discussed in [6], holds the promise of providing mobile computing with highly responsive cloud services, offering scalability and privacy policy enforcement for the Internet of Things, and concealing momentary cloud outages.

Edge intelligence emerges from the integration of edge computing with the high-demand field of AI. In the article outlined by [7], a distinction is made between AI on the edge (artificial intelligence on the edge) and AI for the edge (intelligence-enabled edge computing). The former concentrates on employing established and potent AI technologies to present more optimal solutions for important challenges in edge computing. On the other hand, they later discuss executing the entire process of developing AI models, encompassing model training and inference, directly on the edge. The article offers a comprehensive perspective on this nascent interdisciplinary field, covering fundamental concepts and outlining a research roadmap that provides the necessary background for potential new edge intelligence research projects.

The authors [8] present an in-depth analysis of AI’s contribution to IoT security in edge computing. The research begins with an overview of the current situation and essential definitions. Subsequently, the EC IoT service framework is elucidated, followed by an examination of blockchain and privacy protection for edge-enabled IoT services with AI. The article concludes by highlighting challenges and unanswered questions regarding the utilization of AI in IoT services built on edge computing.

A distinctive AI-enabled multi-access edge computing (MEC) platform supporting IoT applications with computing-equipped unmanned aerial vehicles (UAVs) is explored in [7]. The study employs a game-theoretic model to address the challenge of selecting optimal data offloading techniques from IoT nodes to UAV-mounted MEC servers, considering communication and computation costs. The article demonstrates the existence of at least one Pure Nash Equilibrium (PNE) point, showcasing various operation points based on the results of the Best Response Dynamics (BRD) algorithm or alternative reinforcement learning approaches. In [9], the author discuss into the detailed analysis of the coordinated design of flexible AIoT (Artificial Intelligence of Things) systems across the end-edge-cloud spectrum. The exploration commences by defining core concepts such as edge computing, AI, and the Internet of Things, providing a framework for understanding the basic AIoT architecture. The article then presents a practical AIoT example, showcasing the application of AI in real-world scenarios, and reviews recent developments in AI model inference and training at the network's edge. It concludes by outlining existing challenges and potential directions in this dynamic field.

Edge intelligence has gained special attention for its goal of simplifying the deployment of deep learning (DL) services through edge computing [10]. Furthermore, deep learning, as a representative artificial intelligence (AI) technique, can be incorporated into edge computing frameworks to establish intelligent edges for adaptive and dynamic edge management. This paper presents and explores the following aspects: (1) application scenarios of both edge intelligence and intelligent edge; (2) practical implementation methods and enabling technologies, specifically DL training and inference within a tailored edge computing framework; and (3) challenges and future trends aimed at achieving more pervasive and fine-grained intelligence. This study aims to help readers comprehend the connections among enabling technologies, fostering further research by amalgamating information scattered across the communication, networking, and DL domains.

In the work by [11], three models are proposed: (1) a system-level battery model evaluating energy dissipation in IoT devices; (2) a forward central dynamic and available approach (FCDAA) adjusting running times of sensing and transmission processes in IoT-based portable devices; and (3) a data reliability model for edge AI-based IoT devices over a hybrid Transmission Power Control (TPC) and duty-cycle network. Two important cases for proper industrial platform monitoring, static (e.g., product processing) and dynamic (e.g., vibration and defect diagnostics), are introduced. The suggested FCDAA enhances energy efficiency and battery lifetime with acceptable reliability (0.95) by appropriately tuning duty cycle and TPC in contrast to existing approaches.

The advent of the Internet of Things (IoT) and cloud computing has transformed organizations and people's lifestyles [12]. Technological advancements and digital miniaturization have led to increasingly smarter devices. Applications are shifting from the cloud to the edge due to limitations in the cloud paradigm, including processing complexity, latency in service provisioning, and inefficient resource allocation. Moreover, the integration of artificial intelligence into edge computing has garnered attention for its capacity to accelerate the pace and scope of IoT applications.

Both the computer system and AI communities' express eagerness for a dedicated forum to exchange the latest developments in edge intelligence (EI) research, which is still in its early stages. To address this need, [13] has conducted a comprehensive analysis of current EI research initiatives. Initially, the history and purpose of AI operating at the network edge are examined. The paper then briefly discusses broad architectural frameworks, evolving core technologies, and the training/inference of deep learning models at the network edge. Finally, potential directions for future EI research are explored. The authors anticipate that this survey will attract increasing attention, stimulate engaging debates, and inspire new ideas for EI research.

Edge devices, located close to end users, provide most of the computational power for next-generation IoT applications. These AI-powered devices, equipped with built-in intelligence, can autonomously make decisions in their usage environment. In response to issues about potential threats, [14] proposes a cognitive intrusion security system to safeguard the integrity of search engine results by preventing advertising images from entering the web browser's image database. The suggested framework considers three distinct levels—data collection services, edge computing services, and cloud services—to deliver edge intelligence for web data filtration and detect web spam [15].

To significantly enhance traffic efficiency and road safety, next-generation Vehicle-to-Everything (V2X) networks are envisioned as communication platforms introducing various novel applications, most relying on ultrareliable low-latency communications. The future V2X networks aim to transition from existing connected vehicles, largely forming vehicle-to-vehicle networks, to fully heterogeneous V2X networks supporting autonomous driving. This transition is facilitated by the deep integration of fog computing (FC) and artificial intelligence (AI) technologies. In the survey [16], an innovative FC-based vehicle network design is discussed, accompanied by AI-enabled, fog-assisted V2X use cases that leverage essential FC capabilities and harness the power of AI to facilitate the intended evolution of vehicular networks. Finally, the paper identifies several potential AI-related challenges in FC-based V2X networks for further development.

3 Motivation

The field of AI-based edge computing for next-generation computing systems presents numerous motivations that drive its development and adoption. Here are some key motivating factors:

- *Real-Time Decision Making:* Instant decision-making capabilities are important for next-generation computing systems, particularly to support time-critical applications like autonomous vehicles, industrial automation, and healthcare monitoring. AI-based edge computing strategically positions AI algorithms closer to the data source, facilitating real-time processing and immediate responses, all without dependence on cloud connectivity. This impetus propels the advancement of edge computing to meet the stringent timing requirements inherent in these applications.
- *Latency Reduction:* Numerous applications cannot endure the latency incurred by transmitting data to the cloud for processing and awaiting the response. AI-based edge computing, through conducting AI computations at the edge in close proximity to the data source, mitigates latency and enhances system responsiveness. This reduction in latency holds particular importance for applications that require rapid and seamless

user experiences, including augmented reality, virtual reality, and online gaming.

- *Bandwidth Optimization:* The substantial increase in data generated by IoT devices, sensors, and interconnected systems presents challenges related to bandwidth limitations and data transfer costs. AI-based edge computing addresses these issues by optimizing bandwidth utilization through local data processing and filtering at the edge. Only pertinent or processed information is transmitted to the cloud, diminishing the volume of data transferred and alleviating network congestion.
- *Privacy and Data Security:* With increasing issues about data privacy and security, AI-based edge computing offers a solution by locally processing data on edge devices. This approach minimizes the transmission of sensitive data to the cloud, thereby safeguarding user privacy and ensuring data security. This becomes particularly important in scenarios involving personal health data, confidential business information, or surveillance systems.
- *Edge Intelligence and Autonomy:* AI-based edge computing empowers edge devices with on-board intelligence, enabling them to autonomously make decisions by analyzing real-time data and understanding their context. This impetus propels the evolution of edge computing systems capable of operating independently, adapting to dynamic environments, and diminishing reliance on cloud connectivity, thereby enhancing system autonomy.
- *Offline Operation and Resilience:* Next-generation computing systems frequently function in varied environments, encompassing remote or disconnected areas characterized by limited or unreliable network connectivity. AI-based edge computing empowers edge devices to operate offline or with intermittent connectivity, ensuring system resilience and uninterrupted functionality. This capability proves important in applications related to remote monitoring, disaster management, or critical infrastructure, where continuous operation is imperative.
- *Scalability and Distributed Learning:* AI-based edge computing facilitates distributed learning and collaborative intelligence. Edge devices can collaboratively contribute to AI model training while safeguarding data privacy. This distributed learning approach supports scalable systems capable of continuous model improvement and adaptive

learning, reducing reliance on centralized cloud servers. This impetus propels the advancement of edge computing systems designed to handle large-scale deployments and dynamic learning scenarios.

In summary, the motivations for AI-based edge computing in next-generation systems revolve around the need for real-time decision-making, latency reduction, bandwidth optimization, privacy, data security, edge intelligence, offline operation, and scalability. By addressing these motivations, AI-based edge computing drives the advancement of computing systems to meet the demands of diverse applications, enabling faster, more efficient, secure, and intelligent systems at the edge.

4 Popular Applications Used AI Based Edge Computing Systems in This Smart Era

In the smart era, AI-based edge computing systems are being applied to various domains and industries to enhance efficiency, responsiveness, and intelligence at the edge [17]. Here are some popular applications where AI-based edge computing is being utilized:

- *Autonomous Vehicles:* AI-based edge computing is important for autonomous vehicles as it enables real-time decision-making and quick response times. Edge devices equipped with AI algorithms process sensor data, perform object detection and recognition, and make immediate decisions to ensure safe and efficient navigation on the roads.
- *Industrial Internet of Things (IIoT):* AI-based edge computing is widely used in industrial settings to optimize processes, enhance productivity, and enable predictive maintenance. Edge devices equipped with AI capabilities analyze sensor data, monitor equipment health, and detect anomalies in real-time, enabling proactive maintenance and minimizing downtime.
- *Smart Cities:* AI-based edge computing assumes a pivotal role in smart city applications. Deployed across the city, edge devices gather and process data from diverse sources, like sensors, cameras, and connected devices. At the edge, AI algorithms facilitate real-time analysis of data, supporting applications such as traffic management, environmental monitoring, waste management, and public safety.

- *Healthcare:* In the healthcare industry, AI-based edge computing improves patient monitoring, remote diagnostics, and real-time decision-making. Edge devices equipped with AI algorithms process medical data at the point of care, enabling faster diagnosis, personalized treatment, and reduced dependence on centralized systems.
- *Retail:* AI-based edge computing is utilized in retail environments for various applications, including inventory management, customer analytics, and personalized shopping experiences. Edge devices with AI capabilities analyze in-store data, monitor shelf inventory, track customer behavior, and deliver personalized recommendations, improving operational efficiency and enhancing customer satisfaction.
- *Smart Homes:* AI-based edge computing enhances the capabilities of smart home devices by enabling local processing and intelligent automation. Edge devices equipped with AI algorithms can interpret sensor data, recognize patterns, and autonomously control devices for energy management, home security, and personalized user experiences.
- *Video Surveillance:* AI-based edge computing is employed in video surveillance systems to enhance security and automate video analytics. Edge devices with AI capabilities analyze video feeds in real-time, enabling object detection, facial recognition, and behavior analysis for immediate threat detection and response.
- *Energy Management:* AI-based edge computing is used in energy management systems to optimize energy consumption and improve efficiency. Edge devices equipped with AI algorithms analyze energy usage data, predict demand, and automate control mechanisms, leading to optimized energy utilization and cost savings.
- *Agriculture:* AI-based edge computing assists in precision agriculture by monitoring and managing crops, optimizing irrigation, and detecting disease or pest outbreaks. Edge devices with AI capabilities process sensor data from fields, analyze crop health parameters, and provide real-time information for informed decision-making and proactive interventions.
- *Edge Gaming:* AI-based edge computing is utilized in cloud gaming and mobile gaming scenarios to enhance gameplay experiences. Edge devices with AI algorithms process game data, perform real-time rendering, and enable low-latency interactions, providing immersive and responsive gaming experiences.

Hence, these are the examples of the popular applications where AI-based edge computing is making an important impact in the smart era. As the technology continues to advance, we can expect further expansion into diverse domains, enabling intelligent and autonomous systems at the edge.

5 Current Stage of AI Based Edge Computing System

As of my knowledge, the cutoff in September 2021 is AI-based edge computing systems are already in use and have made important progress. However, it’s important to note that technology is constantly evolving, and advancements may have occurred since then [18–20]. Here’s an overview of the current stage of AI-based edge computing systems:

- *Industry Adoption:* AI-based edge computing has garnered momentum across diverse industries, encompassing automotive, manufacturing, healthcare, retail, and telecommunications. Organizations are leveraging edge computing to facilitate real-time decision-making, diminish latency, and augment operational efficiency.
- *Infrastructure Development:* Companies are investing in developing edge computing infrastructure to support AI-based edge computing systems. This includes the deployment of edge nodes, edge servers, and edge gateways that are equipped with computational resources and AI capabilities to process data at the edge.
- *Edge Device Capabilities:* Edge devices, including IoT devices, cameras, and sensors, are growing in power and capability, enabling them to run AI algorithms locally. They can execute tasks such as image recognition, natural language processing, and predictive analytics without excessive reliance on cloud connectivity.
- *Edge AI Frameworks:* There are several AI frameworks and libraries available that are specifically designed for edge computing. These frameworks optimize AI models for deployment on resource-constrained edge devices, allowing efficient inference and real-time processing.
- *Edge-Cloud Integration:* AI-based edge computing systems are often integrated with cloud computing resources to create a hybrid architecture. This integration enables seamless data transfer between the edge and the cloud, allowing for offloading heavy computation to the cloud for more

complex AI tasks while still maintaining low-latency processing at the edge.
- *Standardization Efforts:* Standardization bodies and industry consortia are working toward establishing common standards for AI-based edge computing. These standards aim to ensure interoperability, data security, and scalability across different edge computing deployments, enabling easier adoption and integration.
- *Research and Development:* There is ongoing research and development in the field of AI-based edge computing. Researchers are exploring new algorithms, techniques, and architectures to optimize AI model training and inference at the edge. This includes federated learning, edge intelligence, and distributed AI approaches.
- *Use Cases and Applications:* AI-based edge computing is being applied to a wide range of use cases, as mentioned earlier. The development and deployment of AI-enabled autonomous vehicles, smart city infrastructure, industrial IoT systems, and healthcare monitoring systems are examples of how AI-based edge computing is already being utilized.

Note that the stage of AI-based edge computing may vary across industries and specific applications. While some industries have already embraced AI-based edge computing, others may still be in the early stages of adoption. However, the continuous advancements in AI, edge computing technologies, and the growing need for real-time, intelligent processing suggest a promising future for AI-based edge computing systems.

---

6 Recent Trends in AI Based Edge Computing System

In recent years, AI-based edge computing systems have witnessed several notable trends. These trends aim to enhance the capabilities and efficiency of edge devices while enabling more intelligent and real-time processing at the network edge [21, 22]. Here are some of the prominent trends in AI-based edge computing systems:

- *Edge AI Chips:* There has been an important development in specialized hardware chips designed specifically for AI processing at the edge. These chips, such as NVIDIA Jetson series, Google Edge TPU, and Intel

Movidius Neural Compute Stick, provide accelerated processing power for AI workloads on edge devices.

- *Federated Learning:* Federated Learning is gaining prominence in AI-based edge computing systems, allowing the training of AI models on distributed edge devices without necessitating the transmission of raw data to a central server. This approach facilitates privacy-preserving machine learning and mitigates latency by processing data locally.
- *Edge-to-Cloud Collaboration:* The collaborative use of AI-based edge devices with cloud computing resources is on the rise. Edge devices manage real-time processing and inference tasks, while the cloud offers additional computational power for training intricate AI models and storing extensive datasets. This hybrid approach strikes a balance, leveraging the advantages of both edge computing and cloud computing.
- *Edge-Native AI Models:* Traditional AI models are often resource-intensive and require important computational power. However, edge devices typically have limited resources. To address this challenge, there is a growing focus on developing edge-native AI models that are specifically designed for efficient execution on edge devices. These models are smaller in size, optimized for low-power consumption, and capable of running on resource-constrained hardware.
- *Edge Intelligence and Autonomy:* Edge devices are evolving to be more intelligent and capable of autonomous decision-making. Through the direct integration of AI algorithms into these devices, they can analyze and interpret data locally, make real-time decisions, and respond swiftly without dependence on cloud connectivity. This trend is particularly pertinent for applications demanding low latency and high reliability, such as autonomous vehicles and industrial IoT.
- *Security and Privacy:* As the number of edge devices proliferates and the processing of sensitive data at the edge rises, security and privacy have emerged as critical issues. AI-based edge computing systems are integrating advanced security measures, including hardware-based encryption, secure bootstrapping, and trusted execution environments, to safeguard data and thwart unauthorized access.

Hence, these trends collectively aim to empower edge devices with enhanced AI capabilities, enabling them to process and analyze data efficiently, make intelligent decisions, and operate autonomously while addressing challenges such as resource constraints, latency, and data privacy.

---

7 Future Research Opportunities with AI Based Edge Computing Systems

AI-based edge computing systems offer numerous exciting research opportunities that can shape the future of edge computing and artificial intelligence [23]. Here are some potential research directions in this field:

- *Edge AI Model Optimization:* Developing advanced techniques to optimize AI models specifically for edge devices is an ongoing research area. This includes model compression, quantization, and pruning techniques that reduce model size and computational requirements while preserving accuracy. Further exploration of model architecture design tailored for edge devices can also lead to more efficient and effective AI models.
- *Energy-Efficient Edge Computing:* Energy efficiency is important for edge devices, especially those powered by batteries or renewable energy sources. Researchers can focus on developing energy-aware AI algorithms and techniques that minimize power consumption without sacrificing performance. This includes optimizing inference and training processes, resource allocation, and scheduling strategies to maximize energy efficiency in edge computing systems.
- *Edge Intelligence and Real-Time Decision-Making:* Exploring the realm of empowering edge devices to make intelligent decisions in real-time is an intriguing avenue of research. Delving into innovative AI algorithms and approaches that facilitate real-time inference and decision-making on resource-constrained edge devices holds the potential to unlock opportunities for applications such as autonomous vehicles, robotics, and smart grids.
- *Federated Learning and Privacy-Preserving Techniques:* Federated Learning has emerged as a promising method for training AI models on distributed edge devices while safeguarding data privacy. Research efforts can concentrate on enhancing the efficiency and scalability of federated learning algorithms, tackling challenges such as communication overhead, model aggregation, and secure aggregation

protocols. Furthermore, the exploration of novel privacy-preserving techniques, such as differential privacy and secure multiparty computation, can further bolster the privacy guarantees in edge computing systems.
- *Trustworthiness and Security of Edge AI:* Ensuring the trustworthiness and security of edge AI systems is important, mainly when dealing with sensitive data and critical applications. Research can explore techniques for detecting and mitigating adversarial attacks on edge devices, developing robust authentication and access control mechanisms, and investigating secure and accountable AI algorithms for edge computing.
- *Edge-Cloud Collaboration and Resource Orchestration:* In the collaboration of edge devices and cloud resources within AI-based systems, research can center on crafting efficient resource orchestration algorithms. These algorithms would optimize task offloading, workload distribution, and resource management between the edge and the cloud. This involves delving into dynamic workload partitioning, load balancing, and task scheduling techniques to attain optimal performance and resource utilization.
- *Domain-Specific Edge AI Applications:* There is vast potential for the development of domain-specific AI applications at the edge. Research endeavors can discuss the application of AI in diverse areas such as healthcare, smart cities, agriculture, manufacturing, and transportation. The creation of specialized AI models, tailored data acquisition techniques, and domain-specific algorithms has the potential to revolutionize these industries and contribute to important societal benefits.

Note that these research opportunities present exciting avenues for advancing AI-based edge computing systems, addressing key challenges, and unlocking the full potential of edge AI in various domains [24].

7.1 AI-IoT Based Edge Computing Systems

IoT devices generate extensive real-time data. Utilizing AI algorithms at the edge facilitates the instantaneous processing and analysis of data, enabling swift decision-making and faster response times. This is particularly important for time-sensitive applications, including industrial automation, smart cities, and autonomous vehicles. Now here are a few other uses (of AI-IoT based Edge computing systems) are:

- *Reduced Latency:* In AI-IoT edge computing systems, data processing takes place locally on edge devices, minimizing the necessity for data transmission to a centralized cloud or data center. This reduction in the need for long-distance data travel decreases latency, making it well-suited for applications demanding real-time or near-real-time responses, such as remote monitoring, predictive maintenance, and emergency response systems.
- *Bandwidth Optimization:* Conducting data processing and filtering at the edge allows AI-IoT systems to minimize the volume of raw data requiring transmission to the cloud or data center. Only pertinent or condensed information is forwarded, optimizing bandwidth utilization and mitigating network congestion. This proves particularly advantageous in situations where bandwidth is restricted or expensive, such as in remote locations or constrained networks.
- *Enhanced Privacy and Security:* AI-IoT edge computing systems effectively tackle privacy and security issues by conducting data processing locally. Rather than sending sensitive or private data to the cloud, edge devices can perform data analysis and extract meaningful information without compromising privacy. This methodology minimizes the exposure of sensitive data to potential security risks and ensures compliance with privacy regulations.
- *Improved Scalability and Reliability:* AI-IoT edge computing systems distribute computational tasks across a network of edge devices, reducing the load on central servers or cloud infrastructure. This enhances scalability, as additional edge devices can be easily added to handle growing workloads. Moreover, edge devices can continue to function even in the event of network disruptions or intermittent connectivity, ensuring reliable operation and reducing dependency on a centralized infrastructure.
- *Context-Aware Decision-Making:* AI algorithms deployed at the edge can use contextual information from IoT devices to make more informed and context-aware decisions. For example, in a smart home system, edge devices can analyze sensor data, user preferences, and environmental conditions to autonomously adjust temperature, lighting, and other parameters based on the occupants’ needs and energy efficiency goals.
- *Edge-Cloud Collaboration:* AI-IoT edge computing systems leverage the capabilities of both edge devices and cloud infrastructure. Edge devices

excel in real-time data processing and local decision-making, while intricate AI training or resource-intensive tasks can be offloaded to the cloud for processing. This hybrid approach provides the strengths of edge computing and cloud computing, resulting in a more efficient and powerful system.

Therefore, AI-IoT-based edge computing systems present vast opportunities for diverse applications, spanning smart homes, smart cities, industrial automation, healthcare monitoring, environmental monitoring, and beyond. The synergy of AI and IoT at the edge unleashes the potential for real-time intelligence, decreased latency, heightened privacy, and enhanced scalability. This transformative combination is reshaping various industries and facilitating innovative solutions.

7.2 AI-Blockchain Based Edge Computing Systems

Edge computing systems based on AI and blockchain integrate artificial intelligence, blockchain technology, and edge computing to establish distributed and secure environments for data processing and decision-making at the network edge [25–27]. The following outlines key aspects and benefits of AI-Blockchain based edge computing systems:

- *Data Integrity and Immutability:* The utilization of blockchain technology offers a decentralized and tamper-resistant ledger for the secure storage and validation of data. Integrating AI with blockchain at the edge enables the secure storage of data processed and generated by edge devices on the blockchain, guaranteeing data integrity and immutability. This is particularly beneficial in applications where maintaining data integrity is important, such as in supply chain management, smart contracts, and critical infrastructure.
- *Trusted Data Sharing and Collaboration:* AI-Blockchain based edge computing systems enable secure and trusted data sharing among multiple edge devices. Blockchain technology facilitates the establishment of trust among participants by providing a transparent and auditable record of data transactions. This enables secure collaboration and data sharing between edge devices, fostering cooperative AI algorithms and applications across distributed edge environments.
- *Data Privacy and Control:* Through AI-Blockchain at the edge, both individuals and organizations gain increased control over their data. Blockchain technology facilitates the implementation of privacy-

preserving mechanisms, including data encryption and access control, ensuring that only authorized parties can access and process the data. This not only strengthens privacy protection but also empowers individuals with greater control over their personal data.
- *Consensus Mechanisms and Governance:* Blockchain technology utilizes consensus mechanisms to validate and agree upon the state of the distributed ledger. In AI-Blockchain based edge computing systems, consensus mechanisms can be used to verify and validate AI models, training data, or decision-making processes across distributed edge devices. This ensures the integrity and trustworthiness of AI operations at the edge.
- *Incentive Mechanisms and Data Monetization:* Blockchain-based edge computing systems can introduce incentive mechanisms to encourage participation and data sharing among edge devices. This enables individuals or organizations to monetize their data by securely sharing it with other participants in the network, creating new economic models and opportunities for data marketplaces and decentralized AI applications.
- *Scalability and Fault Tolerance:* Blockchain technology provides inherent scalability and fault tolerance in distributed systems. By using blockchain at the edge, AI applications can benefit from the decentralized nature of the blockchain network, enabling efficient and fault-tolerant processing and decision-making across a network of edge devices.
- *Auditing and Compliance:* AI-Blockchain based edge computing systems offer enhanced auditing and compliance capabilities. The transparent and immutable nature of the blockchain ledger enables efficient audit trails, ensuring that data processing and AI operations comply with regulatory requirements. This is mainly relevant in industries with strict compliance standards, such as healthcare, finance, and legal sectors.

In summary, AI-Blockchain based edge computing systems provide a robust and secure infrastructure for decentralized AI processing and decision-making at the network edge. The combination of AI, blockchain, and edge computing technologies enables trusted data sharing, data privacy, integrity, and decentralized governance, opening new possibilities for secure and collaborative AI applications in various domains.

7.3 AI-Blockchain-IoT Based Edge Computing Systems

AI-Blockchain-IoT based edge computing systems combine the capabilities of artificial intelligence (AI), blockchain technology, and the Internet of Things (IoT) at the network edge. These systems enable secure and decentralized data processing, intelligent decision-making, and data sharing among IoT devices [26, 28]. Here are some key aspects and benefits of AI-Blockchain-IoT based edge computing systems:

- *Secure and Trusted Data Exchange:* Integrating blockchain technology offers a decentralized and tamper-resistant ledger for recording transactions and data exchanges. When combining AI, blockchain, and IoT at the edge, data produced by IoT devices can be securely stored and exchanged among edge devices through smart contracts. This approach guarantees secure and trusted data sharing within IoT ecosystems, elevating data integrity and eliminating the necessity for centralized intermediaries.
- *Data Privacy and Ownership:* AI-Blockchain-IoT systems empower individuals and organizations to retain control over their data and uphold privacy standards. Leveraging blockchain technology enables the implementation of privacy-preserving mechanisms like encryption and access control. Additionally, AI algorithms can process data locally on edge devices, safeguarding sensitive information from external exposure. This approach strengthens data privacy and ownership, addressing issues in IoT environments where data is frequently shared among multiple stakeholders.
- *Distributed AI Model Training:* Edge devices in an AI-Blockchain-IoT system can collaborate to collectively train AI models using federated learning or distributed learning techniques. Blockchain technology ensures the integrity and transparency of the training process by recording model updates and aggregating them securely. This enables distributed AI model training while preserving data privacy and reducing the need for data transfer to a centralized server.
- *Trust and Accountability:* In AI-IoT edge computing systems, blockchain technology acts as a trust layer. The transparent and immutable characteristics of the blockchain ledger allow participants to verify the authenticity and integrity of data, AI models, and decisions executed at the edge. This instills trust and accountability among system stakeholders, thereby elevating transparency and auditability.

- *Scalability and Fault Tolerance:* AI-Blockchain-IoT systems offer scalability and fault tolerance by using the decentralized nature of blockchain and edge computing. Edge devices can collectively process and analyze data, reducing the load on centralized systems. Additionally, blockchain's consensus mechanisms ensure fault tolerance, enabling the system to continue functioning even if some edge devices become unavailable.
- *Autonomous Decision-Making:* The amalgamation of AI, blockchain, and IoT empowers edge devices to autonomously make intelligent decisions. By locally processing and analyzing real-time data using AI algorithms, leveraging blockchain for data verification and trust, and utilizing smart contracts for automation, edge devices can act independently. This autonomy allows them to make informed decisions based on local context and AI models.
- *Data Marketplace and Monetization:* AI-Blockchain-IoT systems have the potential to establish decentralized data marketplaces, providing a secure platform for IoT device owners to share and monetize their data. With blockchain ensuring transparent and auditable transactions, AI algorithms can analyze and extract valuable information from the data. This creates new avenues for data monetization, offering incentives to data contributors and driving innovation across diverse industries.

Note that AI-Blockchain-IoT based edge computing systems offer a powerful framework for secure, decentralized, and intelligent IoT data processing and decision-making. They provide enhanced data privacy, ownership, trust, and scalability, enabling innovative applications in sectors such as smart cities, healthcare, logistics, and supply chain management.

---

8 Open Issues Toward AI Based Edge Computing Systems

While AI-based edge computing systems offer important advantages, there are still several open issues and challenges that need to be addressed for their widespread adoption and effectiveness [27, 29]. Some of the key open issues include (refer to Fig. 1):

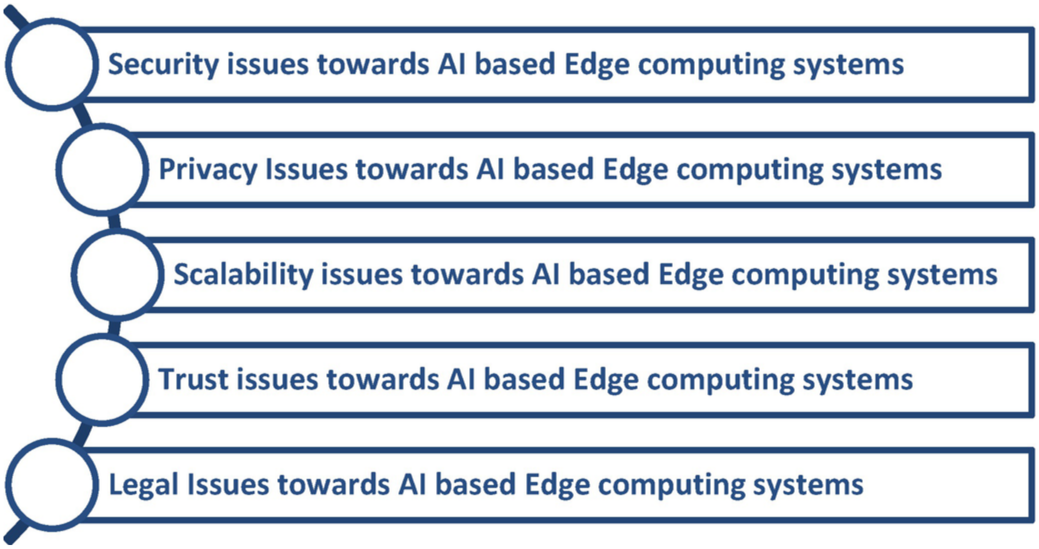

Security issues towards AI based Edge computing systems
Privacy Issues towards AI based Edge computing systems
Scalability issues towards AI based Edge computing systems
Trust issues towards AI based Edge computing systems
Legal Issues towards AI based Edge computing systems

Fig. 1 Issues toward artificial intelligence based edge computing

- Resource constraints: Creating AI models and algorithms that operate effectively within the limitations of edge devices, including restricted computational resources, storage capacity, and power supply, poses a persistent challenge. To address this, there is a need for efficient model compression, lightweight algorithms, and optimization techniques that empower AI processing on resource-constrained edge devices.
- Data heterogeneity and quality: Edge devices in IoT environments generate data with varying formats, quality, and characteristics. Handling data heterogeneity and ensuring data quality pose challenges for AI-based edge computing systems. Developing robust data preprocessing, cleaning, and fusion techniques that can handle diverse data sources and ensure data quality is important for accurate and reliable AI processing.
- Edge-cloud coordination and collaboration: The collaboration between edge devices and cloud resources in AI-based edge computing systems necessitates effective coordination. Achieving a balance in workload distribution, optimizing data transfer, and ensuring smooth integration demand efficient coordination mechanisms and communication protocols. Addressing challenges related to the efficient allocation of resources and decision-making for task offloading becomes important in these systems.
- Security and privacy: Ensuring security and privacy is paramount in AI-based edge computing systems. Given that edge devices handle sensitive data, safeguarding data security, preventing unauthorized access, and ensuring data privacy are critical considerations. The development of

robust security mechanisms, encryption techniques, access control, and privacy-preserving algorithms becomes essential to effectively addressing these issues.
- Trust and transparency: AI algorithms deployed at the edge should be trustworthy and transparent to gain user and stakeholder confidence. Ensuring the transparency of AI models, training data, and decision-making processes is challenging, especially in distributed edge environments. Incorporating explainability and interpretability into AI models and developing auditing mechanisms for edge-based AI systems are ongoing research areas.
- Scalability and manageability: Scaling AI-based edge computing systems to a large number of devices and managing them efficiently is a challenge. As the number of edge devices increases, managing software updates, model deployments, and system configurations becomes complex. Developing scalable management frameworks, device orchestration mechanisms, and remote monitoring techniques is important to support large-scale deployments.
- Regulatory and ethical issues: AI-based edge computing systems need to comply with regulatory requirements and ethical guidelines. Ensuring fairness, avoiding bias in AI models, and addressing ethical issues related to data collection, privacy, and decision-making are critical. Incorporating ethics and regulatory compliance into the design and deployment of AI-based edge systems is an ongoing research area.

Hence, these open issues require interdisciplinary research efforts involving AI, edge computing, networking, cybersecurity, and privacy. Solving these challenges will enable the development of robust, efficient, secure, and trustworthy AI-based edge computing systems that can unlock the full potential of edge devices and revolutionize various application domains.

8.1 Security Issues Toward AI Based Edge Computing Systems

AI-based edge computing systems present unique security challenges that must be addressed to guarantee the integrity, confidentiality, and availability of both data and services [30, 31]. Here are some key security issues for AI-based edge computing systems:

- Device and endpoint security: AI-based systems' edge devices are susceptible to diverse security threats, encompassing physical tampering, unauthorized access, malware, and software vulnerabilities. Safeguarding the security of edge devices necessitates implementing secure boot processes, regular software updates, and robust access control mechanisms to thwart unauthorized access and preserve the integrity of the AI system.

- Data security and privacy: AI-based edge computing systems frequently handle sensitive and personal data, encompassing health records, financial information, or user behavior data. Safeguarding the confidentiality and privacy of this data is paramount. Utilizing encryption techniques, secure data transfer protocols, and privacy-enhancing technologies like differential privacy can fortify the protection of data, both in transit and at rest.
- Secure communication: Securing communication channels between edge devices, cloud services, and other system components is imperative to thwart eavesdropping, tampering, and unauthorized access. The implementation of secure communication protocols, such as Transport Layer Security (TLS), coupled with secure key exchange mechanisms, guarantees the integrity and confidentiality of data during transmission.
- Authentication and access control: AI-based edge computing systems require robust authentication mechanisms to validate the identity of devices, users, and services. Implementing access control mechanisms is important to guarantee that solely authorized entities can access and manipulate the AI system. This involves employing multi-factor authentication, secure credential management, and role-based access control (RBAC) mechanisms.
- Adversarial attacks on AI models: AI models deployed at the edge face vulnerability to adversarial attacks where adversaries manipulate input data to deceive AI algorithms, resulting in incorrect decisions or compromised system behavior. Utilizing strategies such as robust model training, anomaly detection, and model verification aids in detecting and mitigating adversarial attacks in AI-based edge systems.

To maintain system resilience and fault tolerance, AI-based edge computing systems must withstand failures, disruptions, and malicious attacks. Implementing redundancy, fault tolerance mechanisms, and backup strategies is important to ensure system availability and recoverability. This involves deploying techniques like distributed consensus algorithms, backup edge devices, and comprehensive disaster recovery plans.

- Regulatory compliance: Ensuring compliance with regulations and standards concerning data protection, privacy, and cybersecurity is imperative for AI-based edge computing systems. Adhering to pertinent regulations, including the General Data Protection Regulation (GDPR), implementing robust data handling and consent mechanisms, and conducting routine security audits are important steps to maintain compliance and foster trust with users and stakeholders.

Hence, these security issues require a holistic approach that encompasses secure system design, encryption, secure communication protocols, access control mechanisms, anomaly detection, and ongoing monitoring and updates. Implementing routine security assessments, vulnerability testing, and incident response plans is essential to identifying and addressing security risks in AI-based edge computing systems.

8.2 Privacy Issues Toward AI Based Edge Computing Systems

AI-based edge computing systems raise important privacy issues due to the collection, processing, and storage of sensitive data at the network edge [32]. Here are some key privacy issues for AI-based edge computing systems:

- Data collection and consent: To address the collection of personal and sensitive data in AI-based systems, it is important to implement proper consent mechanisms and adhere to privacy regulations. Transparent data collection practices, explicit user consent mechanisms, and clear privacy policies are essential in ensuring compliance and building trust with users.
- Data minimization and purpose limitation: AI-based edge systems must adhere to the principles of data minimization and purpose limitation, ensuring that only essential data is collected and used solely for specified purposes. The collection of excessive data poses increased privacy risks. Restricting data collection to what is strictly necessary for the intended AI processing and avoiding unnecessary retention of data are essential measures to mitigate privacy issues.
- Data anonymization and pseudonymization: To protect privacy, AI-based edge systems can utilize strategies like data anonymization and pseudonymization. Anonymization involves removing or encrypting personally identifiable information (PII), while pseudonymization substitutes direct identifiers with pseudonyms. These measures contribute

to the protection of individual privacy and mitigate the risk of re-identification.

- Secure data storage and transmission: The secure storage and transmission of data in AI-based edge systems are essential for preserving privacy. Strong encryption mechanisms, secure protocols, and secure storage practices should be employed to protect data both at rest and in transit. This includes secure key management, secure storage containers, and secure communication protocols.
- Differential privacy: Differential privacy serves as a privacy-enhancing technique applicable to AI-based edge systems, safeguarding individual privacy during data analysis. By introducing noise or perturbation to the data, differential privacy prevents the re-identification of individuals while still enabling meaningful analysis and AI processing.
- User access control and transparency: AI-based edge systems must empower users with control over their data and offer transparency in its utilization. Users should possess the capability to manage data preferences and exercise access and deletion rights. Transparent information regarding data processing, employed AI algorithms, and the intended purposes of data usage should be readily available to users.
- Third-party data sharing: In AI-based edge systems that engage in third-party data sharing for collaborative AI processing or data monetization, it is important to establish stringent data sharing agreements. Ensuring secure data sharing exclusively with trusted and authorized entities is imperative. Thorough consideration should be given to potential privacy risks associated with the sharing of data with external parties.
- Privacy impact assessments: Performing privacy impact assessments (PIAs) proves instrumental in recognizing and mitigating privacy risks within AI-based edge systems. PIAs encompass the evaluation of data flow, data handling practices, privacy safeguards, and the potential privacy risks inherent in the system. This proactive approach empowers organizations to address privacy issues and implement suitable privacy controls.

Hence, privacy issues in AI-based edge computing systems require a privacy-by-design approach, ensuring privacy issues are embedded into the system architecture, data processing workflows, and user interactions. Compliance with relevant privacy regulations, transparent data practices, and regular privacy audits are important steps in protecting individuals' privacy in AI-based edge computing environments.

8.3 Scalability Issues Toward AI Based Edge Computing Systems

Scalability is an important consideration in the realm of AI-based edge computing systems. Edge computing involves the localized processing of data and execution of computations in proximity to the data source, deviating from the traditional reliance on centralized cloud servers [31]. This strategy proves particularly advantageous for AI applications demanding low latency, real-time decision-making, and minimized network bandwidth. Here are some scalability challenges that can arise in AI-based edge computing systems:

- Limited computational resources: Edge devices, such as IoT devices, often have limited processing power, memory, and storage capacities. Running resource-intensive AI algorithms directly on these devices can be challenging due to their constrained nature. Scaling AI models to accommodate a large number of edge devices with varying capabilities becomes a complex task.
- Bandwidth constraints: Edge devices typically operate with limited network connectivity. Transmitting large amounts of data to the cloud for processing can quickly consume network bandwidth, leading to increased latency and reduced system performance. Scaling AI models while minimizing data transfer becomes important to overcome these constraints.
- Model size and complexity: AI models, especially deep learning models, can be large and computationally expensive. Deploying such complex models on edge devices can be impractical or inefficient due to the limited computational resources available. Developing lightweight AI models that balance accuracy and resource requirements is essential for scalability in edge computing systems.
- System management and orchestration: Managing a large number of edge devices, each running its own AI models, can be challenging. Coordinating the deployment, updates, and monitoring of AI models across diverse edge devices requires efficient system management and

orchestration. Scaling the management infrastructure to handle a growing number of edge devices is critical.

- Data privacy and security: Securing distributed edge devices that process sensitive data is a critical aspect of edge computing systems. Safeguarding data privacy and security at the edge is essential for protecting user information and preventing unauthorized access. An important challenge lies in scaling security measures, including encryption, authentication, and access control, across a multitude of edge devices.

Further, to address such scalability, etc., issues, several strategies can be used:

- Model optimization: Optimizing AI models to reduce their size, complexity, and computational requirements can make them more suitable for deployment on edge devices. Techniques like model compression, quantization, and knowledge distillation can help create lightweight models without essential loss in accuracy.
- Federated learning: Federated learning facilitates collaborative training of AI models across various edge devices while maintaining the data on those devices. By minimizing the necessity for data transfer, this approach enables the scalability of AI models without centralizing data, contributing to enhanced privacy.
- Edge-cloud collaboration: Integrating the capabilities of edge computing and cloud computing forms a hybrid architecture, offering a solution to scalability challenges. This hybrid model involves delegating resource-intensive tasks to the cloud, while real-time processing occurs at the edge, providing a balanced approach to address scalability requirements within resource constraints.
- Edge orchestration platforms: Using edge orchestration platforms can simplify the management, deployment, and monitoring of AI models on edge devices. These platforms provide centralized control and automation, enabling efficient scaling and updates across a distributed edge infrastructure.
- Edge caching and pre-processing: Caching frequently used AI models or pre-processing data at the edge can reduce the need for repeated computation and data transfer. This approach can improve scalability by minimizing the computational and network resource requirements.

In summary, addressing scalability issues in AI-based edge computing systems requires a combination of optimized AI models, efficient system management, data privacy measures, and collaboration between edge and cloud resources.

8.4 Trust Issues Toward AI Based Edge Computing Systems

Trust plays a pivotal role in AI-based edge computing systems. Processing data and making decisions closer to the data source, a characteristic of edge computing, can give rise to issues regarding privacy, security, reliability, and transparency [26]. Here are some trust issues associated with AI-based edge computing systems:

- Data privacy: Edge devices frequently handle sensitive data, including personal information and proprietary business data, introducing a risk of interception or compromise during transmission or storage. Trust issues emerge as users express issues about data privacy, potential misuse, and unauthorized access to their information.
- Security vulnerabilities: Edge devices, especially IoT devices, are often resource-constrained and may lack robust security mechanisms. These devices can become potential entry points for cyberattacks or malicious activities. The trustworthiness of AI-based edge systems can be undermined if they are susceptible to security breaches that compromise data integrity or expose vulnerabilities.
- Algorithmic bias and fairness: AI models deployed on edge devices may exhibit biases if they are trained on biased data or if the models themselves are flawed. Biased decisions made by AI systems at the edge can lead to unfair outcomes and erode trust. Ensuring fairness and transparency in AI algorithms used in edge computing is important to maintaining trust.
- Lack of interpretability: Deep learning models, which are commonly used in AI applications, can be complex and difficult to interpret. The lack of transparency in how these models make decisions raises issues about accountability and trust. Understanding why an AI system arrived at a particular decision or recommendation is essential, especially in critical applications like healthcare or autonomous vehicles.
- Reliability and robustness: Edge devices may operate in challenging environments with fluctuating network connectivity, limited power supply, or varying environmental conditions. The reliability and

robustness of AI-based edge systems can be questioned if they fail to perform consistently or exhibit unpredictable behavior. Trust is built when edge systems reliably deliver accurate results, even under adverse conditions.

- Vendor lock-in and interoperability: Deploying AI-based edge systems may involve using proprietary technologies or vendor-specific solutions. Locking into a specific vendor can limit flexibility, scalability, and interoperability, leading to issues about future adaptability and potential dependence on a single provider. Trust is enhanced when systems are built on open standards and promote interoperability.

To address these trust issues, several measures can be taken:

- Privacy-preserving techniques: Utilizing privacy-preserving techniques, such as data encryption, secure communication protocols, and differential privacy, can safeguard sensitive data and mitigate privacy issues.
- Robust security measures: Implementing robust security mechanisms, such as device authentication, access control, and encryption, is important to fortify edge devices and secure the data they handle. Consistent application of security updates and patches helps mitigate vulnerabilities and ensures the reliability of the system.
- Explainable AI: Promoting interpretability and explainability in AI models used at the edge can increase trust. Techniques like explainable machine learning and model-agnostic interpretability can provide information about how AI systems arrive at decisions, enabling better accountability and transparency.
- Fairness and bias mitigation: Addressing algorithmic biases involves using diverse and representative training data, implementing bias detection and correction techniques, and employing fairness-aware learning algorithms. These measures contribute to fostering fair outcomes and cultivating trust in the system.
- Robustness testing: Conducting thorough testing and validation of AI models in various edge environments helps ensure their reliability, resilience, and robustness. This includes evaluating model performance under different network conditions, system failures, and environmental factors.
- Standards and certifications: Developing industry standards, certifications, and best practices for AI-based edge computing systems can help establish trust and promote interoperability. Compliance with

recognized standards and certification programs can provide assurance to users and stakeholders.

Hence, building trust in AI-based edge computing systems requires a holistic approach that addresses data privacy and security.

8.5 Legal Issues Toward AI Based Edge Computing Systems

AI-based edge computing systems raise various legal issues that need to be considered [32–36]. Here are some key legal issues associated with AI-based edge computing systems:

- Data protection and privacy: Edge computing systems, which process and store data locally on edge devices, must adhere to data protection regulations like the General Data Protection Regulation (GDPR) in the European Union. It is important to handle personal data lawfully and securely, ensuring users are informed about the collected data, its purpose, and their rights related to their data.
- Data ownership and control: Determining ownership and control of data in AI-based edge computing systems can be complex. Clear contractual agreements and terms of service should outline the ownership and control rights of data collected and processed by edge devices. Users should have control over their data and understand how it is used, shared, and retained.
- Consent and transparency: In the realm of data protection and privacy regulations, obtaining informed consent from users is essential. AI-based edge computing systems ought to furnish transparent details about their data processing activities, encompassing the types of data collected, the processing purpose, and any potential sharing with third parties.
- Liability and accountability: Determining liability in AI-based edge computing systems can be challenging due to the distributed nature of processing and decision-making. If an AI system deployed on an edge device causes harm or makes erroneous decisions, it may raise questions of accountability. Clear contractual agreements, appropriate risk allocation, and liability provisions should be in place to address potential liabilities.
- Intellectual property rights: AI models and algorithms used in edge computing systems can be subject to intellectual property rights. Organizations should ensure that they have the necessary rights to use the AI models, either through ownership or appropriate licensing

agreements. Intellectual property issues become mainly important when deploying third-party AI models on edge devices.

- Ethical and legal issues in AI decision-making: AI-based edge systems may make autonomous decisions that have legal and ethical implications. It is important to ensure that the decision-making processes of AI models comply with legal requirements, such as non-discrimination laws or sector-specific regulations. The transparency and explainability of AI algorithms become important to assess the legality and ethics of the decisions made.
- Compliance with sector-specific regulations: Various industries are subject to distinct regulations and standards governing the use of AI-based edge computing systems. In healthcare, adherence to medical device regulations and patient privacy laws is imperative. A thorough understanding and compliance with industry-specific regulations are essential to ensuring legal adherence.

To address these legal issues, it is recommended to:

- Conduct a thorough legal analysis: Assess the legal implications of deploying AI-based edge computing systems in your specific jurisdiction and industry. Understand the relevant regulations, guidelines, and legal requirements that apply to data protection, privacy, intellectual property, liability, and other relevant areas.
- Adopt privacy-by-design and security measures: Integrate privacy and security features into the design and development of AI-based edge computing systems. Implement privacy-enhancing technologies, employ data anonymization methods, enforce access controls, and utilize encryption to safeguard data and ensure alignment with relevant regulations.
- Obtain legal advice: Consult legal professionals with expertise in data protection, privacy, intellectual property, and technology law. They can provide guidance on legal compliance, contract drafting, liability issues, and intellectual property rights related to AI-based edge computing systems.
- Conduct regular audits and assessments: Periodically review and assess the compliance of AI-based edge systems with legal requirements. Conduct data protection impact assessments (DPIAs), security audits, and vulnerability assessments to identify and mitigate potential legal risks.

- Stay informed about evolving regulations: Stay updated with legal developments and changes in regulations related to AI, data protection, and privacy.

---

9 Critical Challenges Toward AI Based Edge Computing Systems

AI-based edge computing systems face several critical challenges that need to be addressed for their successful deployment and operation [34]. Here are some key challenges (refer Fig. 2):

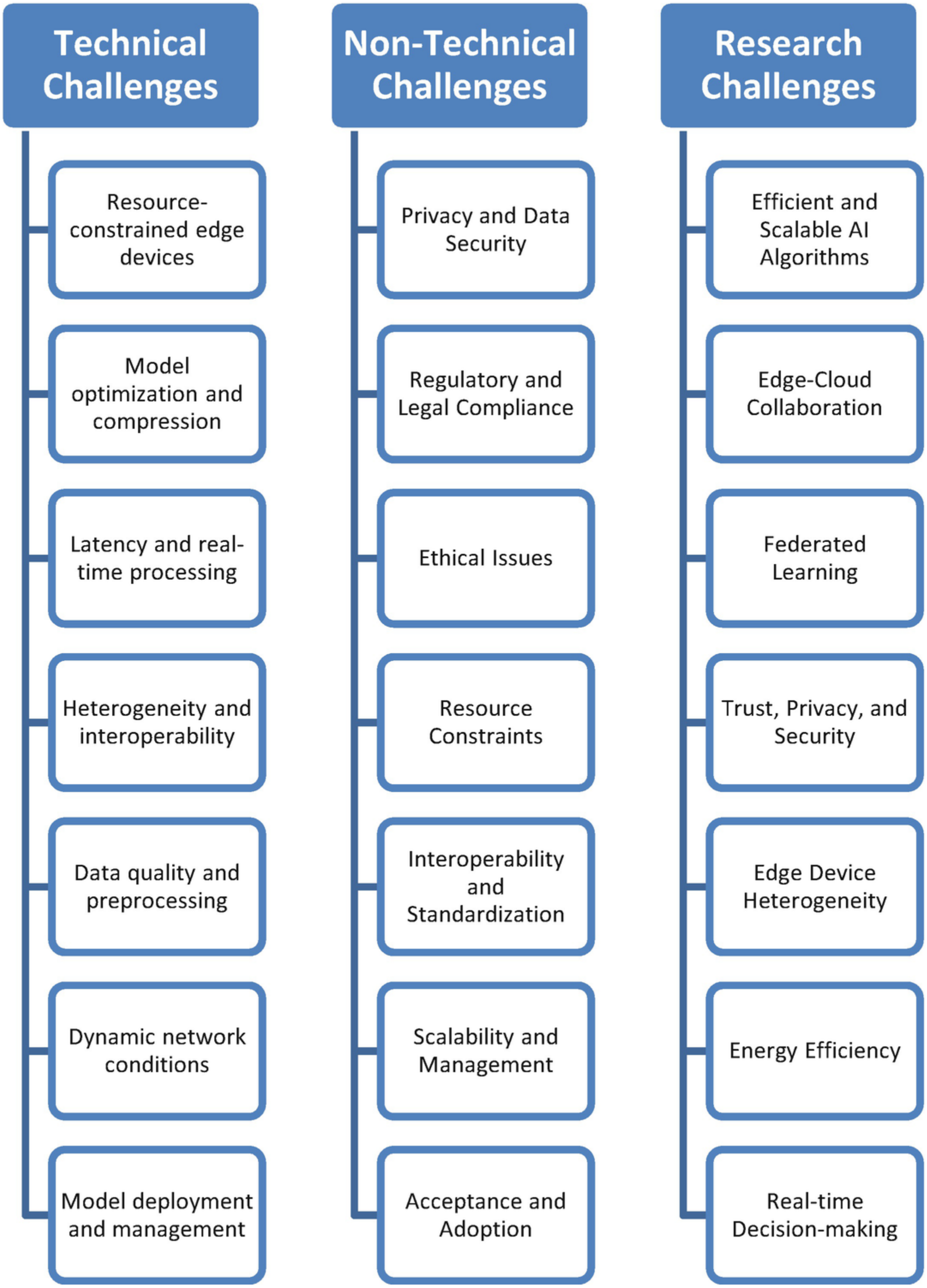

Technical Challenges
Resource-constrained edge devices
Model optimization and compression
Latency and real-time processing
Heterogeneity and interoperability
Data quality and preprocessing
Dynamic network conditions
Model deployment and management
Non-Technical Challenges
Privacy and Data Security
Regulatory and Legal Compliance
Ethical Issues
Resource Constraints
Interoperability and Standardization
Scalability and Management
Acceptance and Adoption
Research Challenges
Efficient and Scalable AI Algorithms
Edge-Cloud Collaboration
Federated Learning
Trust, Privacy, and Security
Edge Device Heterogeneity
Energy Efficiency
Real-time Decision-making

Fig. 2 Challenges in AI-based edge computing

- Limited computational resources: Edge devices commonly face constraints in processing power, memory, and storage capabilities. Executing AI algorithms with high resource demands on such devices presents an important challenge. The optimization of AI models for lightweight efficiency, balancing resource limitations with accuracy, becomes important in this context.

- Bandwidth and network limitations: Edge devices frequently function with restricted or sporadic network connectivity. Transmitting substantial data to and from the cloud for processing may prove impractical or slow. To address bandwidth and network constraints, it is essential to minimize data transfer, employ efficient data compression techniques, and leverage local processing capabilities at the edge.
- Data quality and diversity: AI algorithms rely on high-quality, diverse, and representative data for training and inference. Collecting and curating such data at the edge can be challenging due to limited data availability, data noise, and data distribution heterogeneity across different edge devices. Ensuring data quality, diversity, and relevance for effective AI model training becomes critical.
- Real-time decision-making: Some AI applications demand instantaneous or near-instantaneous decision-making, such as autonomous vehicles or industrial automation. Nevertheless, edge devices might possess restricted processing capabilities, and intricate AI algorithms could introduce latency. Effectively addressing this challenge involves developing and optimizing AI models and algorithms specifically for low-latency and real-time processing at the edge.
- Security and privacy issues: Edge devices often process sensitive data, and ensuring data privacy and security becomes paramount. Edge devices may have vulnerabilities, and securing them against attacks is critical. Additionally, privacy issues arise when data is processed locally on edge devices. Implementing robust security measures, encryption, access controls, and privacy-preserving techniques is essential to addressing these challenges.
- Scalability and management: Managing many distributed edge devices and AI models can be complex. Coordinating model deployment, updates, monitoring, and system management at scale requires efficient orchestration, automation, and scalability of management infrastructure. Ensuring seamless integration and interoperability across heterogeneous edge devices is also a challenge.
- Model training and updating: Training AI models at the edge presents challenges due to constraints in computational resources and data

availability. The application of federated learning techniques, which involve collaborative model training across multiple edge devices without sharing raw data, proves beneficial in overcoming these challenges. Another critical challenge lies in efficiently updating and retraining models while minimizing disruption to edge devices.

- Interoperability and standardization: Edge computing encompasses a variety of edge devices, communication protocols, and AI frameworks, creating a diverse ecosystem. Ensuring interoperability and standardization among these heterogeneous components can pose challenges. The development of open standards, interfaces, and frameworks becomes important to promote interoperability and facilitate seamless integration.
- Ethical and regulatory issues: AI-based edge computing systems present ethical challenges concerning algorithmic bias, fairness, and transparency. Complying with legal and regulatory requirements, including data protection, privacy, and industry-specific regulations, becomes challenging in a distributed edge environment. Tackling these ethical and regulatory issues necessitates meticulous design, accountability, and transparency in AI-based edge systems.

Note that overcoming these challenges requires a combination of technological advancements, algorithmic optimizations, infrastructure enhancements, regulatory frameworks, and collaboration among stakeholders, including researchers, industry experts, policymakers, and standardization bodies.

9.1 Technical Challenges Toward AI Based Edge Computing Systems

AI-based edge computing systems face several technical challenges that need to be addressed for their effective implementation. Here are some key technical challenges:

- Resource-constrained edge devices: Edge devices, such as IoT devices, frequently possess restricted computational power, memory, and energy resources. Executing sophisticated AI algorithms on these devices can be computationally intensive and energy-draining. Developing lightweight and efficient AI models capable of functioning within the limitations of edge devices is an important challenge.

- Model optimization and compression: Optimizing AI models for deployment on resource-constrained edge devices is imperative in edge computing systems. Utilizing model compression techniques, including quantization, pruning, and knowledge distillation, facilitates the reduction of model size and computational demands while preserving acceptable accuracy levels. Striking the optimal balance between model size, complexity, and performance poses a notable technical challenge.
- Latency and real-time processing: Edge computing seeks to facilitate real-time or near-real-time decision-making. Nonetheless, the restricted computational resources of edge devices, coupled with the imperative to process data locally, pose challenges related to latency. The optimization of AI algorithms to achieve low-latency and real-time processing at the edge is important, particularly in applications such as autonomous vehicles, robotics, and industrial automation.
- Heterogeneity and interoperability: Edge computing environments encompass a variety of edge devices featuring distinct hardware architectures, operating systems, and communication protocols. Overcoming technical challenges related to ensuring interoperability and smooth integration among these heterogeneous devices is imperative. The development of standardized interfaces, protocols, and frameworks that facilitate efficient communication and interoperability stands as a key requirement.
- Data quality and preprocessing: Edge devices may encounter noisy or incomplete data due to factors such as sensor limitations or communication errors. Ensuring data quality and performing preprocessing tasks, such as data cleaning, filtering, and feature extraction, at the edge can be challenging. Developing robust data preprocessing techniques that address the limitations of edge devices is important for maintaining accurate AI models.
- Dynamic network conditions: Edge devices often operate in dynamic network conditions with intermittent connectivity or limited bandwidth. This poses challenges for data transmission and synchronization between edge devices and the cloud. Developing adaptive algorithms and protocols that can handle varying network conditions, handle data buffering, and prioritize data transfer is important.
- Model deployment and management: Managing AI models deployed on many distributed edge devices can be complex. Challenges include

model deployment, updates, monitoring, and maintenance across edge devices. Efficient model version control, over-the-air updates, and remote management capabilities are necessary to ensure smooth operation and scalability of AI-based edge systems.
- Edge-cloud coordination: Edge computing systems often require coordination and interaction between edge devices and the cloud. Optimizing the workload distribution and task offloading between the edge and the cloud while minimizing latency and maximizing resource utilization is a technical challenge. Developing efficient edge-cloud coordination mechanisms and decision-making algorithms is important.
- Security and privacy: Securing edge devices against threats like data breaches, tampering, and unauthorized access is important. Addressing the technical challenge of safeguarding AI models, data, and communications at the edge involves the implementation of robust security measures. This includes incorporating encryption, access controls, and secure communication protocols to ensure comprehensive protection.

Note that addressing these technical challenges requires advancements in AI algorithms, model optimization techniques, edge device architectures, communication protocols, and system management frameworks. Collaboration between researchers, industry experts, and technology providers is important for overcoming these challenges and realizing the full potential of AI-based edge computing systems.

9.2 Non-technical Challenges Toward AI Based Edge Computing Systems

While AI-based edge computing systems offer numerous benefits, they also face several non-technical challenges. These challenges are primarily related to organizational, societal, and ethical aspects. Here are some of the non-technical challenges associated with AI-based edge computing systems:

- Privacy and Data Security: The localization of data processing in edge computing introduces issues regarding privacy and data security. AI algorithms executing on edge devices may necessitate access to sensitive information, including personal details, location data, and health records. It becomes imperative to establish secure practices for data transmission, storage, and processing to safeguard user privacy and thwart unauthorized access.

- Regulatory and Legal Issues: Deploying AI-based edge computing systems often involves compliance with various regulatory and legal frameworks. These systems may need to adhere to data protection regulations, industry-specific standards, and ethical guidelines. Organizations must navigate these complex compliance requirements to ensure legal operation and avoid penalties or legal consequences.
- Resource Constraints: Edge computing systems operate with limited computational power, memory, and energy resources compared to centralized cloud environments. This constraint can pose challenges for deploying sophisticated AI models and running resource-intensive algorithms on edge devices. Optimization techniques, model compression, and efficient resource allocation strategies need to be developed to address these limitations.
- Interoperability and Standardization: The proliferation of edge devices originating from different manufacturers, coupled with the variety of AI frameworks, may lead to challenges in interoperability. The absence of standardized practices across hardware platforms, communication protocols, and AI frameworks complicates the creation of scalable and interoperable AI-based edge computing systems. Initiating standardization endeavors is essential to facilitate smooth integration and collaboration among diverse edge devices and platforms.
- Scalability and Management: Managing a large number of distributed edge devices can be complex. Ensuring the scalability of AI-based edge computing systems, including device discovery, configuration, monitoring, and software updates, requires efficient management frameworks. Additionally, handling the heterogeneity of edge devices, including varying capabilities and configurations, adds further complexity to system management.
- Acceptance and Adoption: Convincing stakeholders, including users, organizations, and policymakers, about the benefits of AI-based edge computing can be a challenge. Lack of awareness, trust issues, and resistance to change might impede the widespread adoption of these systems. Demonstrating the value proposition, addressing issues, and promoting education and awareness about AI-based edge computing are important for fostering acceptance and adoption.

Hence, addressing these non-technical challenges requires a multidisciplinary approach involving collaboration between technologists, policymakers, industry leaders, and researchers to ensure the successful deployment and sustainable growth of AI-based edge computing systems.

9.3 Research Challenges Toward AI Based Edge Computing Systems

Research challenges in the field of AI-based edge computing systems encompass several areas that require further exploration and innovation. Here are some key research challenges:

- Efficient and Scalable AI Algorithms: Developing AI algorithms specifically tailored for edge devices is essential. These algorithms should be resource-efficient and capable of running on devices with limited computational power and memory. Research efforts are needed to design novel algorithms that strike a balance between accuracy and efficiency while considering the constraints of edge computing environments. Additionally, developing scalable algorithms that can handle increasing data volumes and diverse edge device deployments is important.
- Edge-Cloud Collaboration: Investigating the optimal collaboration between edge devices and cloud infrastructure is an ongoing research challenge. Determining the division of computation between edge devices and the cloud, as well as designing efficient data transfer and synchronization mechanisms, is important for maximizing the benefits of edge computing while using the power of the cloud. Research is needed to explore methods for workload distribution, latency optimization, and seamless integration between the edge and cloud components.
- Federated Learning: Federated learning enables training machine learning models collaboratively across distributed edge devices while preserving data privacy. Research in this area involves developing efficient and secure federated learning algorithms that can handle the resource constraints of edge devices, address communication limitations, and maintain data privacy and security. Improving the convergence speed, robustness, and fairness of federated learning methods are ongoing research challenges.

- Trust, Privacy, and Security: Ensuring trust, privacy, and security in AI-based edge computing systems is a critical research challenge. Developing robust mechanisms to protect user data and prevent unauthorized access, while enabling secure communication and computation, is essential. Research efforts should focus on addressing privacy issues, developing secure data sharing protocols, and designing methods to detect and mitigate security vulnerabilities and attacks in edge environments.
- Edge Device Heterogeneity: Edge computing environments consist of diverse devices with varying computational capabilities, network connectivity, and energy resources. Research is needed to handle the heterogeneity of edge devices, including developing adaptive algorithms that can adapt to different device capabilities and efficiently utilize available resources. Exploring techniques such as model partitioning, dynamic resource allocation, and intelligent task scheduling can help optimize performance across heterogeneous edge devices.
- Energy Efficiency: Energy efficiency is an important issue in edge computing due to the limited power resources of edge devices. Research efforts should focus on developing energy-aware AI algorithms and optimization techniques that minimize the energy consumption of edge devices while maintaining acceptable performance levels. Exploring energy harvesting techniques and energy-aware task offloading strategies are important research directions in this domain.
- Real-time Decision-making: Many edge computing applications demand instantaneous decision-making capabilities. There is a requirement for research to create streamlined and low-latency AI algorithms capable of processing and analyzing data in real-time on edge devices. Employing techniques like lightweight deep learning models, hardware acceleration, and optimizations tailored for edge environments can contribute to achieving real-time responsiveness.
- Edge System Management: Managing many distributed edge devices is a complex task. Research challenges include developing efficient device discovery mechanisms, scalable system management frameworks, and intelligent software update and maintenance strategies. Additionally, investigating self-configuring and self-healing techniques for edge devices can enhance the robustness and reliability of edge computing systems.

In summary, addressing these research challenges will contribute to the advancement of AI-based edge computing systems, enabling their broader adoption, and realizing their full potential in various application domains.

---

10 Lesson Learned for AI Based Edge Computing Systems for Next Generation Computing Environment

As AI-based edge computing systems continue to evolve, several valuable lessons have been learned that can guide the development and deployment of these systems in the next generation computing environment [32, 35]. Here are some key lessons:

- Consider Edge-Cloud Collaboration: The next generation computing environment should emphasize the collaboration and integration between edge devices and the cloud. Rather than viewing edge and cloud as separate entities, a hybrid approach that considers the strengths of both should be adopted. This collaboration enables efficient resource utilization, improved scalability, and optimized decision-making capabilities.
- Design for Heterogeneity: The next generation computing environment is likely to comprise a wide range of edge devices with varying capabilities, such as sensors, wearables, and IoT devices. Designing AI-based edge computing systems that can accommodate this heterogeneity is important. The systems should be flexible, adaptive, and capable of handling diverse device types and configurations.
- Privacy-Preserving Techniques: Privacy is being most important in edge computing systems owing to the proximity of sensitive data to the edge devices. The forthcoming systems must accord priority to privacy-preserving techniques like federated learning, secure computation protocols, and differential privacy. Such approaches facilitate collaborative learning while safeguarding user data and ensuring adherence to privacy regulations.
- Energy Efficiency and Sustainability: Energy efficiency is a critical issue in the next generation computing environment. Edge devices are often resource-constrained and may operate on limited battery power. Incorporating energy-efficient AI algorithms, optimizing resource

allocation, and exploring energy harvesting techniques will contribute to sustainable and environmentally friendly edge computing systems.

- Robust Security Measures: Security should be a top priority in the next generation computing environment. Edge devices are vulnerable to security threats, making it essential to implement robust security measures. Secure communication protocols, encryption techniques, and intrusion detection systems are necessary to safeguard data and prevent unauthorized access.
- Standardization and Interoperability: The next generation of computing environments should emphasize the establishment of standards and the promotion of interoperability among various edge devices and platforms. Engaging in standardization endeavors facilitates the smooth integration, efficient communication, and collaboration between diverse edge devices, fostering scalability and interoperability across the ecosystem.
- User-Centric Design: The next generation computing environment should prioritize user-centric design principles. Understanding user needs, preferences, and usability requirements will enhance the acceptance and adoption of AI-based edge computing systems. Emphasizing intuitive interfaces, personalized experiences, and ease of use will contribute to positive user experiences.

Hence, by addressing these lessons, the next generation computing environment can foster the development of robust, efficient, and user-centric AI-based edge computing systems that use the full potential of edge devices while seamlessly integrating with cloud infrastructure.

11 Conclusion

In conclusion, AI-driven edge computing has become imperative for the future of computing systems, offering distinct advantages and capabilities. By positioning AI algorithms and processing capabilities closer to data sources, edge computing delivers real-time analysis, reduced latency, heightened privacy, and enhanced efficiency. This approach facilitates intelligent decision-making at the edge, surpassing limitations seen in centralized cloud computing and supporting diverse applications across various domains.

The necessity of AI-enabled edge computing is underscored by several factors. Firstly, the explosive growth of data from connected devices and sensors requires efficient localized processing to mitigate network congestion and latency. Edge computing presents a scalable, distributed architecture adept at handling the volume, velocity, and variety of data at the network edge. Secondly, the ascendance of AI and machine learning applications demands low-latency, real-time processing to enable intelligent decision-making. Edge computing systems, capable of processing and analyzing data locally, empower swift responses in applications like autonomous vehicles, industrial automation, smart cities, and healthcare monitoring.

Thirdly, issues related to privacy and data security advocate for decentralized data processing. Edge computing systems retain sensitive data on local devices, mitigating the risk of data breaches and unauthorized access. This aspect gains importance in sectors such as healthcare, finance, and personal devices where data privacy is paramount. Lastly, the future computing system necessitates energy-efficient solutions to address environmental challenges and optimize resource utilization. Edge computing, by minimizing the need for data transmission to the cloud, reduces energy consumption and promotes sustainable computing practices.

AI-driven edge computing signifies a pivotal shift in the computing landscape. Merging the potency of AI algorithms with the agility and proximity of edge devices, it delivers intelligent, real-time capabilities.

References

1. Kimura, J., Shibasaki, H.: Recent advances in clinical neurophysiology. In: Proceedings of the 10th International Congress of EMG and Clinical Neurophysiology, pp. 10–15. October 15–19, Kyoto, Japan (1995)
2. Cao, K., Liu, Y., Meng, G., Sun, Q.: An overview on edge computing research. IEEE Access. **8**, 85714–85728 (2020) [Crossref]
3. Varghese, B., Wang, N., Barbhuiya, S., Kilpatrick, P., Nikolopoulos, D.S.: Challenges and opportunities in edge computing. In: 2016 IEEE International Conference on Smart Cloud (SmartCloud), pp. 20–26. IEEE (2016) [Crossref]
4. Mao, Y., You, C., Zhang, J., Huang, K., Letaief, K.B.: A survey on mobile edge computing: the communication perspective. IEEE Commun Surv Tutor. **19**(4), 2322–2358 (2017) [Crossref]

5. Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L.: Edge computing: vision and challenges. IEEE Internet Things J. **3**(5), 637–646 (2016)
[Crossref]
6. Shi, W., Dustdar, S.: The promise of edge computing. Computer. **49**(5), 78–81 (2016)
[Crossref]
7. Satyanarayanan, M.: The emergence of edge computing. Computer. **50**(1), 30–39 (2017)
[Crossref]
8. Deng, S., Zhao, H., Fang, W., Yin, J., Dustdar, S., Zomaya, A.Y.: Edge intelligence: the confluence of edge computing and artificial intelligence. IEEE Internet Things J. **7**(8), 7457–7469 (2020)
[Crossref]
9. Xu, Z., Liu, W., Huang, J., Yang, C., Jiawei, L., Tan, H.: Artificial intelligence for securing IoT services in edge computing: a survey. Secur. Commun. Netw. **2020**, 1–13 (2020)
10. Fragkos, G., Tsiropoulou, E.E., Papavassiliou, S.: Artificial intelligence enabled distributed edge computing for Internet of Things applications. In: 2020 16th International Conference on Distributed Computing in Sensor Systems (DCOSS), pp. 450–457. IEEE (2020)
[Crossref]
11. Chang, Z., Liu, S., Xiong, X., Cai, Z., Guoqing, T.: A survey of recent advances in edge-computing-powered artificial intelligence of things. IEEE Internet Things J. **8**(18), 13849–13875 (2021)
[Crossref]
12. Wang, X., Han, Y., Leung, V.C.M., Niyato, D., Yan, X., Chen, X.: Convergence of edge computing and deep learning: a comprehensive survey. IEEE Commun Surv Tutor. **22**(2), 869–904 (2020)
[Crossref]
13. Sodhro, A.H., Pirbhulal, S., Victor, H.C., Albuquerque, D.: Artificial intelligence-driven mechanism for edge computing-based industrial applications. IEEE Trans. Industr. Inform. **15**(7), 4235–4243 (2019)
[Crossref]
14. Sivaganesan, D.: Design and development AI-enabled edge computing for intelligent-IoT applications. J. Trend. Comput. Sci. Smart Technol. (TCSST). **1**(02), 84–94 (2019)
15. Zhou, Z., Chen, X., Li, E., Zeng, L., Luo, K., Zhang, J.: Edge intelligence: paving the last mile of artificial intelligence with edge computing. Proc. IEEE. **107**(8), 1738–1762 (2019)
[Crossref]
16. Makkar, A., Ghosh, U., Sharma, P.K.: Artificial intelligence and edge computing-enabled web spam detection for next generation IoT applications. IEEE Sensors J. **21**(22), 25352–25361 (2021)
[Crossref]

17. Rihan, M., Elwekeil, M., Yang, Y., Huang, L., Chen, X., Selim, M.M.: Deep-VFog: when artificial intelligence meets fog computing in V2X. IEEE Syst. J. **15**(3), 3492–3505 (2020) [Crossref]

18. Yu, S., Chen, X., Zhou, Z., Gong, X., Di, W.: When deep reinforcement learning meets federated learning: intelligent multitimescale resource management for multiaccess edge computing in 5G ultradense network. IEEE Internet Things J. **8**(4), 2238–2251 (2020) [Crossref]

19. Dhelim, S., Kechadi, T., Chen, L., Aung, N., Ning, H., Atzori, L.: Edge-enabled metaverse: the convergence of metaverse and mobile edge computing. arXiv preprint arXiv:2205.02764. (2022)

20. Li, E., Zeng, L., Zhou, Z., Chen, X.: Edge AI: on-demand accelerating deep neural network inference via edge computing. IEEE Trans. Wirel. Commun. **19**(1), 447–457 (2019) [Crossref]

21. Jiang, K., Sun, C., Zhou, H., Li, X., Dong, M., Leung, V.C.M.: Intelligence-empowered mobile edge computing: framework, issues, implementation, and outlook. IEEE Netw. **35**(5), 74–82 (2021) [Crossref]

22. Gill, S.S.: Quantum and blockchain based Serverless edge computing: a vision, model, new trends and future directions. Internet Technol. Lett. **2021**, e275 (2021)

23. Kubiak, K., Dec, G., Stadnicka, D.: Possible applications of edge computing in the manufacturing industry—systematic literature review. Sensors. **22**(7), 2445 (2022) [Crossref]

24. McEnroe, P., Shen, W., Liyanage, M.: A survey on the convergence of edge computing and AI for UAVs: opportunities and challenges. IEEE Internet Things J. **9**(17), 15435 (2022) [Crossref]

25. Calo, S.B., Touna, M., Verma, D.C., Cullen, A.: Edge computing architecture for applying AI to IoT. In: 2017 IEEE International Conference on Big Data (Big Data), pp. 3012–3016. IEEE (2017) [Crossref]

26. Yang, R., Richard Yu, F., Si, P., Yang, Z., Zhang, Y.: Integrated blockchain and edge computing systems: a survey, some research issues and challenges. IEEE Commun Surv Tutor. **21**(2), 1508–1532 (2019) [Crossref]

27. Tyagi, A.K., Kukreja, S., Richa, Sivakumar, P.: Role of blockchain technology in smart Era: a review on possible smart applications. J. Inf. Knowl. Manag. (2024). https://doi.org/10.1142/S0219649224500321 [Crossref]

28. Nair, M.M., Mishra, A.K., Tyagi, A.K.: Fog computing and edge computing: open issues, critical challenges and the road ahead for future. In: Proceedings of the 2023 Fifteenth International Conference on Contemporary Computing (IC3-2023), pp. 66–76. Association for Computing Machinery, New York (2023). https://doi.org/10.1145/3607947.3607962

[Crossref]

29. Nair, M.M., Tyagi, A.K.: Chapter 11: AI, IoT, blockchain, and cloud computing: The necessity of the future. In: Pandey, R., Goundar, S., Fatima, S. (eds.) Distributed Computing to Blockchain, pp. 189–206. Academic Press (2023). ISBN 9780323961462. https://doi.org/10.1016/B978-0-323-96146-2.00001-2
[Crossref]

30. Liu, Y., Peng, M., Shou, G., Chen, Y., Chen, S.: Toward edge intelligence: multiaccess edge computing for 5G and Internet of Things. IEEE Internet Things J. **7**(8), 6722–6747 (2020)
[Crossref]

31. Bandi, A.: A review towards AI empowered 6G communication requirements, applications, and technologies in mobile edge computing. In: 2022 6th International Conference on Computing Methodologies and Communication (ICCMC), pp. 12–17. IEEE (2022)
[Crossref]

32. Dinh, D.-L., Nguyen, H.-N., Thai, H.-T., Le, K.-H.: Towards AI-based traffic counting system with edge computing. J. Adv. Transp. **2021**, 1–15 (2021)
[Crossref]

33. Kochovski, P., Gec, S., Stankovski, V., Bajec, M., Drobintsev, P.D.: Trust management in a blockchain based fog computing platform with trustless smart oracles. Futur. Gener. Comput. Syst. **101**, 747–759 (2019)
[Crossref]

34. Deng, C., Fang, X., Wang, X., Law, K.: Software orchestrated and hardware accelerated artificial intelligence: toward low latency edge computing. IEEE Wirel. Commun. **29**(4), 110–117 (2022)
[Crossref]

35. Wu, C., Peng, Q., Xia, Y., Jin, Y., Zhentao, H.: Towards cost-effective and robust AI microservice deployment in edge computing environments. Futur. Gener. Comput. Syst. **141**, 129–142 (2023)
[Crossref]

36. Lapegna, M., Balzano, W., Meyer, N., Romano, D.: Clustering algorithms on low-power and high-performance devices for edge computing environments. Sensors. **21**(16), 5395 (2021)
[Crossref]

OceanofPDF.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024

S. Pal et al. (eds.), *IoT Edge Intelligence*, Internet of Things: Technology, Communications and Computing
https://doi.org/10.1007/978-3-031-58388-9_4

Artificial Intelligence-Based IoT-Edge Environment for Industry 5.0

Souvik Pal[1, 2, 3] ✉
(1) Department of Management Information Systems, Saveetha College of Liberal Arts And Sciences, Saveetha Institute of Medical And Technical Sciences, Chennai, India
(2) Department of Computer Science and Engineering, Sister Nivedita University (Techno India Group), Kolkata, India
(3) Sambalpur University, Sambalpur, India

✉ **Souvik Pal**
Email: souvikpal22@gmail.com

Keywords Internet of Things – Artificial intelligence – Edge computing – Sensor – Machine learning – Deep learning – Industry 5.0 – Cyber-physical system

1 Introduction

The IoT-edge [1] environment has emerged as a new paradigm within the rapidly growing landscape of the Internet of Things (IoT). This paradigm is characterized by the convergence of physical objects, digital connection, and cognitive data processing.

The IoT-edge environment represents a significant transition from centralized computing architectures that prioritize the cloud toward a decentralized architecture. Fundamentally, this environment utilizes the computational and storage capacities of edge devices, including gateways, cellular RANs, micro data centers, mobile devices among others, to perform data processing and analysis in close proximity to the origin of the data. The term IoT-edge environment encompasses a fluid and progressive ecosystem situated within the wider domain of the Internet of Things (IoT). This concept denotes a significant juncture where tangible devices, sensors, and actuators engage with computational resources and data processing capabilities in the periphery of a network, in greater proximity to both the origin of data generation and the data consumers [2]. In the conventional Internet of Things (IoT) paradigm, data is commonly conveyed to centralized cloud servers for the purpose of processing and analysis. Nevertheless, inside the IoT-edge environment, a substantial proportion of data processing occurs in close proximity to the origin of data collection [4].

The convergence of artificial intelligence (AI) and the Internet of Things (IoT) at the edge has become a significant factor in the dynamic environment of Industry 5.0, bringing about revolutionary changes and revolutionizing operational practices across several industries. The integration of advanced technology in this context offers the potential for unparalleled levels of automation, efficiency, and intelligence in industrial operations. Industry 5.0 endeavors to provide a dynamic and adaptive setting by incorporating AI algorithms into the periphery of IoT networks. This integration intends to optimize operations and enable prompt decision-making processes. The fundamental concept of an AI-driven Internet of Things (IoT) environment for Industry 5.0 involves the utilization of sophisticated machine learning models and algorithms to perform data processing directly at the point of origin. This approach minimizes the reliance on centralized processing and improves the system’s ability to promptly respond to stimuli. This allows enterprises to leverage the abundance of data produced by Internet of Things (IoT) devices in real time, thereby accessing valuable insights that can be used to enhance operational efficiency. This shift in paradigm is a departure from conventional industrial systems, when the processing of data and decision-making frequently took place within centralized cloud settings. In contrast, Industry 5.0 incorporates the principle of edge computing, facilitating the localization of essential data processing in closer proximity to the data origin. This approach effectively minimizes latency and strengthens the overall resilience of the system.

The aggregation of artificial intelligence (AI) and the Internet of Things (IoT) at the edge has a wide range of potential applications in the industry, including but not limited to predictive maintenance, quality control, autonomous manufacturing, and intelligent supply chain management. Furthermore, this technology facilitates the prompt adjustment of industries to evolving circumstances, enhancing their ability to withstand and respond effectively to dynamic market requirements.

This introductory statement establishes the foundation for a thorough examination of the diverse aspects pertaining to an artificial intelligence-driven Internet of Things (IoT)-edge ecosystem within the context of Industry 5.0. By fostering collaboration among academics, industry, and technology providers, our goal is to conduct an in-depth exploration of the theoretical underpinnings, practical applications, and prospective ramifications of this revolutionary approach within the industrial sphere. Collectively, our main efforts are directed toward fostering innovation, enhancing operational efficiency, and shaping the trajectory of Industry 5.0.

1.1 Framework for IoT-Edge Environment

IoT-edge computing refers to the operational approach of performing data processing within IoT networks in immediate proximity to the data source. The aforementioned objective is accomplished by employing edge devices and edge gateways, which undertake the task of filtering and processing data in a localized manner, rather than solely depending on a central server or cloud server for data processing. The workflow of IoT-edge environment denotes the sequence of interrelated procedures and phases entailed in the functioning and administration of Internet of Things (IoT) devices and applications situated at the periphery of a network. The operational sequence of an Internet of Things (IoT)-edge environment has several stages, which comprise data collecting, preprocessing, analysis, decision-making, and, optionally, action initiation. The utilization of this methodology has great importance in Internet of Things (IoT) applications, especially in cases where the processing of data in real time or with minimal latency is of utmost significance. In such scenarios, the feasibility of transmitting data to a remote cloud server for analysis may be hindered by latency-related challenges or considerations pertaining to privacy.

The essential elements and phases in the workflow of an Internet of Things (IoT)-edge environment may encompass:

- *Data Acquisition:* The process of data acquisition involves the collection of data from the surrounding environment through the utilization of sensors and Internet of Things (IoT) devices. The information that could be included in this context encompasses a diverse array of factors, such as temperature, humidity, motion, or other relevant characteristics, which may vary based on the specific application being considered.
- *Data Preprocessing:* Data preprocessing is the use of various techniques to raw data obtained from sensors in order to enhance its quality and suitability for analysis. This often includes filtering, cleaning, and transforming the data to ensure its accuracy and reliability prior to conducting any meaningful analysis. This step is necessary to guarantee that the data is in an appropriate format for subsequent processing.
- *Local Analysis:* Within the context of an edge environment, it is frequently possible to carry out preliminary data analysis in a decentralized manner. This may entail the execution of algorithms or models directly on the edge device or gateway.
- *Decision-Making:* Decision-making can be facilitated at the periphery, utilizing the analyzed data, hence obviating the necessity of transmitting data to a centralized server. The importance of this is particularly significant in applications that necessitate immediate answers.
- *Data Transmission (Optional):* In certain scenarios, there may arise a necessity to transfer data to a centralized cloud or server for the purpose of subsequent processing, long-term retention, or interaction with other systems.
- *Security and Privacy Measures:* The implementation of security and privacy measures is of utmost importance in order to guarantee the protection of data at the edge. The aforementioned security measures encompass encryption protocols, access control mechanisms, and safeguards designed to prevent unauthorized access or manipulation.
- *Local Storage (Optional):* The potential inclusion of local storage is contingent upon the particular application, as there may arise a necessity to retain data within a localized environment for a specified duration prior to its transmission or subsequent processing.
- *Integration with Cloud or Central Server (Optional):* Optional integration with a cloud or central server allows for the consolidation of data and insights obtained from edge devices. This integration facilitates

long-term storage, advanced analytics, and the merging of data collected from many edge devices.

- *Feedback Loop (Optional):* The feedback loop, if implemented, allows for the utilization of insights obtained from the edge environment to optimize operations, make adjustments to algorithms, or enhance the overall system performance in certain applications.

The operational process of an Internet of Things (IoT)-edge environment presents significant flexibility and may be customized to meet the precise demands and limitations of a particular application. The framework offers a robust structure for facilitating real-time, contextually aware decision-making in various Internet of Things (IoT) applications.

Figure 1 deals with the framework for IoT-edge environment; the steps for dataflow are as follows:

Step 1:	Data dispatched from IoT-edge Devices to Edge Gateway.
Step 2:	IP based Network goes from Edge Gateway to Edge Server. Edge servers are located closer to the end-users or devices they serve. They are typically deployed at the edge of the network, closer to the data sources.
Step 3:	Edge server is responsible for Realtime Data Processing, Data Caching, Data Buffering, Data Optimization, Data Analysis etc.
Step 4:	The processed data is directed to Cloud Layer.
Step 5:	Those processed data are processed by Cloud Service Providers as per user requirements.
Step 6:	Cloud Data centres are there for data storage drives, virtual servers, and network service.
Step 7:	End users access the resultant data either from Edge servers for low-latency processing needs or from cloud servers for scenarios involving vast data storage and complex computations.

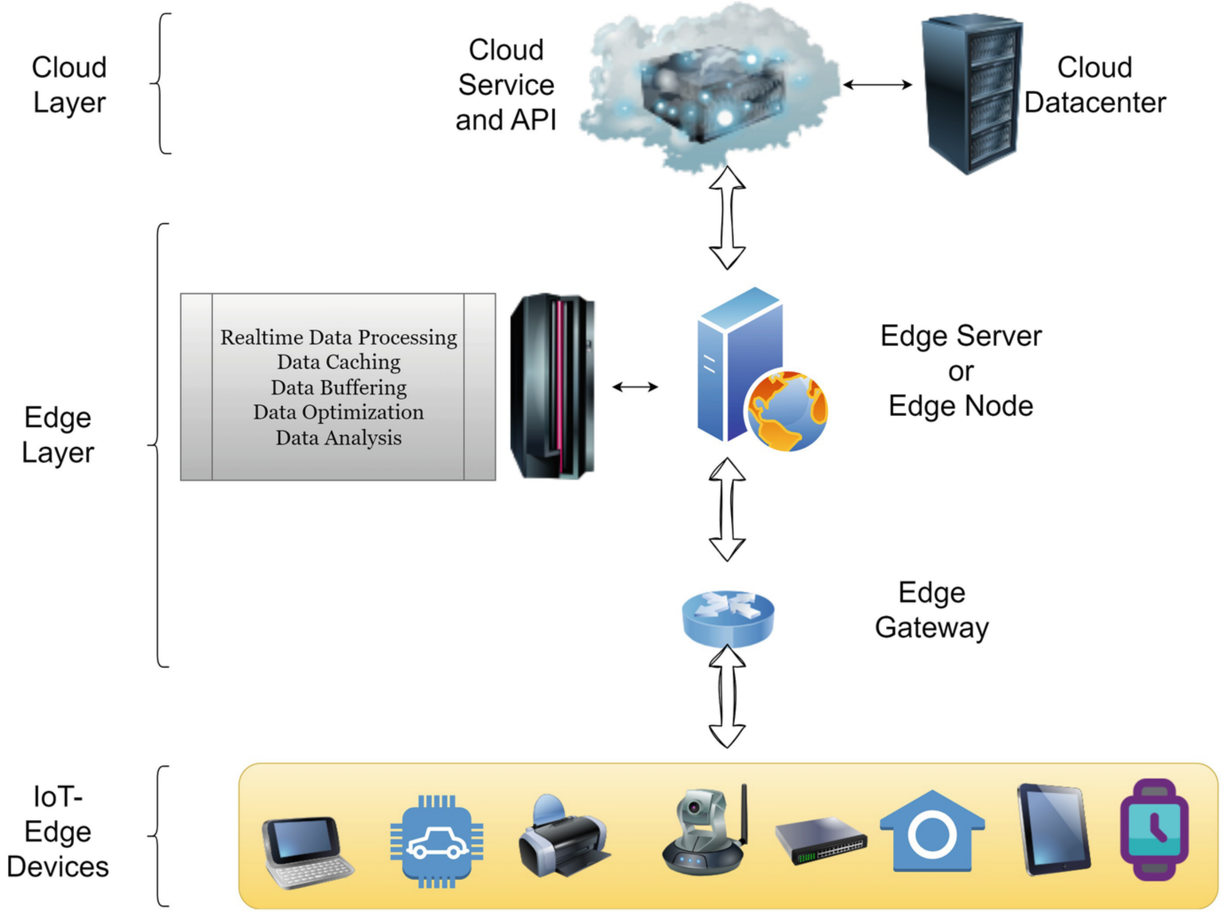

Cloud Layer
Cloud Service and API
Cloud Datacenter
Edge Layer
Realtime Data Processing
Data Caching
Data Buffering
Data Optimization
Data Analysis
Edge Server or Edge Node
Edge Gateway
IoT-Edge Devices

Fig. 1 Framework for IoT-edge environment

1.2 Edge Computing in IoT

Broadly speaking, the concept of the Internet of Things (IoT) encompasses a diverse array of interconnected devices that produce substantial volumes of data. The primary obstacle resides in the effective acquisition and analysis of this data at appropriate junctures in order to convert it into meaningful insights. Currently, Internet of Things (IoT) systems heavily depend on substantial data operations, wherein big data plays a pivotal role in facilitating IoT applications by managing large volumes of data produced through extensive sensing and information stimulation within the IoT ecosystem [38–44]. The acquisition of unstructured, multidimensional data in IoT necessitates the need for further analysis in order to extract meaningful information. The swift advancement of diverse technologies has positioned the Internet of Things (IoT) as the forthcoming technological revolution. However, it introduces challenges in terms of data storage, processing, and analytical capacities. The Internet of Things (IoT) utilizes real-time applications that generate a constant stream of data, which presents issues for data storage in different settings. The resolution of this matter necessitates the implementation of supplementary data centers for the purpose of effectively managing the data acquired from Internet of Things (IoT) devices. One such strategy involves the utilization of platform-as-a-service applications to facilitate the transfer of this information to the cloud. Cloud computing, offered by reputable firms like Google Cloud, Amazon Web Services, and IBM Cloud Analytics, provides benefits in terms of efficiency, capacity, and flexibility for the storage and utilization of sensing data information via the Internet. Nevertheless, the integration of cloud computing into the Internet of Things (IoT) gives rise to novel difficulties. Fog or edge computing emerges as a more expeditious alternative to cloud computing, effectively mitigating the aforementioned concerns. In contrast to the conventional approach of centralizing computational activities in the cloud, fog/edge computing expands processing and storage capacities to devices situated at the periphery of the system. Nodes or objects that possess fog computing characteristics have the ability to effectively manage data storage, computation, and diverse network connectivity at any location within the network. This allows for the seamless connection of Internet of Things (IoT) devices with associated applications.

In general, the data obtained from Internet of Things (IoT) devices is commonly transmitted to appropriate destinations in order to facilitate subsequent analysis in accordance with the specific needs of the application [38–45]. Fog computing and edge computing enable the prompt processing of high-priority data on nodes in close proximity to the relevant Internet of Things (IoT) devices, resulting in the production of substantial empirical evidence. In contrast, data with low priority can be transmitted to communal nodes for further analysis and examination. Despite the numerous advantages it offers, the integration of Internet of Things (IoT) with fog/edge computing presents several obstacles, namely in terms of establishing fog/edge computation and effectively assigning adequate resources to IoT devices. Internet of Things (IoT) devices frequently require a limited range of services, underscoring the importance of optimizing the connection, processing, and storage capabilities of each fog/edge service node. The work of managing assigned resources for fog/edge nodes in an effective manner is a multifaceted endeavor, highlighting the importance of striking a balance between several elements like energy usage, node cost, and service availability. Furthermore, it is imperative to consider the importance of addressing safety and confidentiality problems within the context of fog or edge computing architecture.

2 Industry 5.0

In the twenty-first century, a significant transformation has occurred across various fields, leading to a widespread adoption of digitalization. Nevertheless, it is acknowledged that organizations have challenges when attempting to implement digitalization strategies that involve the integration of artificial intelligence (AI), Internet of Things (IoT), and industry 4.0 technologies. In addition to the aforementioned technologies, it is anticipated that the next phase of the Industrial Revolution, commonly referred to as Industry 5.0, will emerge in the near future [28–37]. The concept of Industry 5.0 emerged in the early part of 2015. It has been referred to as the fifth industrial revolution and has had a significant impact on various sectors, particularly in the realm of everyday business operations. This impact can be attributed to the rapid pace of industrial advancements, technological enhancements, and the integration of human processes [38–41].

The initial phase of industrialization, also referred to as the first Industrial Revolution or Industry 1.0, commenced toward the conclusion of the eighteenth century. This period was characterized by the widespread use of mechanized systems in many industries, which relied on sources of power such as coal, human labor, water, and steam. The onset of the second industrial revolution, often known as Industry 2.0, occurred in the latter part of the nineteenth century. This period was characterized by the widespread adoption of electrical energy as a means of facilitating mass production [42–45]. The emergence of the telephone, mass manufacturing, telegraph, implementation of assembly lines, and mechanization are among the notable characteristics of Industry 2.0. The onset of the third industrial revolution, also known as Industry 3.0, may be traced back to the early twentieth century. The integration of computerization and microelectronic skills was introduced into the industrial sector. The achievement of a more advanced level of automation through the utilization of robotics, information technology, and microprocessors is largely interconnected with projects from the twentieth century, particularly in the field of information and communication technology (ICT). The third industrial revolution has facilitated advancements in various domains such as computer-integrated manufacturing, computer-aided processing planning, computer-aided design, and flexible manufacturing systems. In the early twenty-first century, the advent of the fourth industrial revolution, commonly referred to as Industry 4.0, introduced significant advancements in production through the integration of cyber-physical systems (CPS). Industry 4.0 was primarily distinguished by the integration of cyber-physical systems (CPS), cloud computing, big-data analytics, augmented reality, Internet of Things (IoT), simulation, and intelligent devices. This implies a strong emphasis on the complete digitalization process and the integration of digital industrial ecosystems through the pursuit of fully integrated solutions [43–47]. Moreover, the study primarily emphasizes Internet of Things (IoT) devices that establish connections with industrial facilities.

Industry 5.0 places significant emphasis on the collaboration between humans and various types of machinery. This signifies that the fifth industrial revolution is primarily focused on the development of innovative interfaces that facilitate interaction between humans and machines. The primary objective of Industry 5.0 is to enhance and elevate the advancements achieved in Industry 4.0 to a more advanced stage. This introduces the notion of collaborating robots, sometimes referred to as cobots. The integration of cobots has shown to be successful in meeting the demands of contemporary firms that specialize in the production of customized products [43–47]. Therefore, Industry 5.0 is being implemented in several fields such as manufacturing and medicine, due to advancements in manufacturing techniques, software tools, the Internet of Things, and robotics. This platform offers customers the opportunity to engage in mass customization through collaborative efforts among various organizations on a global scale. Technology advances often fail to take into account the fundamental aspects of organizational revolution, hence necessitating a focus on consumer objectives. In order to achieve the objectives of customers, industry 5.0 adheres to a specific set of principles: Mass customization refers to the provision of personalized products or services to clients, taking into consideration factors such as pricing and comfort. The business approach is centered around prioritizing customer objectives and addressing obstacles to business growth through the process of reengineering. Additionally, there is a strong focus on green computing practices and the consideration of environmental factors. Cyber-physical systems (CPS) is a field that focuses on developing intelligent systems that optimize the interaction between humans and machines, with the aim of maximizing the benefits derived from combining human intelligence with machine intelligence [39–41, 43–47].

Industry 5.0 is expected to provide guidance and resolutions for the challenges related to the elimination of human labor from various manufacturing processes, as discussed earlier. Nevertheless, the implementation of cutting-edge technologies is important in order to enhance the productivity and efficiency of the manufacturer operating within the industry 5.0 framework [39–41, 43–47]. The challenges associated with the assessment and monitoring of manufacturing sites have been brought to light by the advancements in digital twin technology, particularly in the context of intelligent autonomous schemes. This has led to the emergence of difficulties in the valuation of multiscale modeling and simulation. Visualization tools are of very importance in the present context as they play a critical part in the development of policies for the management and customization of authentic products, as well as the creation of product outlines. The utilization of sensor nodes has experienced a significant surge in several domains, ranging from smart homes to autonomous manufacturing cobots and distributed intelligent systems, hence leading to an enhanced need for miniature sensor data interoperability. The aforementioned intelligent sensor nodes has the capability to perceive and gather unprocessed data from the physical environment, hence serving as an indispensable resource for the forthcoming industrial revolution. Nevertheless, the study problem of energy optimization, expeditious and efficient customization procedures, the choice of a local agent for data pre-processing, and the development of highly modeled distributed intelligence in the context of IoT and Industry 4.0 remain unresolved. The combination of virtual reality with digital twin technology has become increasingly viable due to the continual expansion of big data and the advancement of AI-based collaborative robots, enabling the creation of more authentic and lifelike digital replicas. It enables industry professionals to effectively minimize waste in process flow and system design. Therefore, the utilization of advanced visualization techniques in conjunction with digital twin technology is expected to significantly enhance the productivity and efficiency across various industries. The implementation of real-time trackers is expected to enhance the efficiency of real-time production tracking, hence streamlining the process of fulfilling customers’ sales orders by integrating them with manufacturing orders and supplemental materials. Virtual training is a method that can be employed to facilitate the learning of a certain task in a simulated environment, particularly when the trainee and trainer are situated in separate geographical places. This form of training significantly reduces expenses and time for all parties involved. Intelligent autonomous systems, namely artificial intelligence models, play a significant role in the autonomous operation of production lines within the manufacturing industry. The utilization of contemporary artificial intelligence (AI)-based machine learning (ML) and deep learning (DL) models has proven to be highly impactful in effecting modifications inside intelligent systems and solutions, hence enhancing their ability to facilitate decision-making processes. Transfer learning policies are utilized to effectively and gradually implement the aforementioned schemes in a secure manner within the context of Industry 5.0.

The combination of deep learning (DL) and reinforcement learning (RL) techniques, along with the utilization of graphics processing unit (GPU)-based computation, has demonstrated significant promise in replicating rudimentary visual perception and sensory capabilities. However, in order to achieve advanced performances in the context of Industry 5.0, significant enhancements in cobot (collaborative robot) proficiencies are necessary. In a previous paper [48], we have already discussed about the transition from Industry 1.0 to Industry 5.0 in Table 1.

Table 1 Timeline of Industry 1.0 to Industry 5.0

Phase	Period	Description	Identification by	Key point
Industry 1.0	1780–1870	The process of industrial manufacturing utilizing steam and water-powered machinery	Mechanization Water and stream	First mechanical Loom
Industry 2.0	1870–1970	The utilization of electrical energy in mass production	Electrification Division of labor Mass production	First assembly line
Industry 3.0	1970–2011	The integration of electronic and IT systems has facilitated the	Automation Electronics	The first programmable logic controller

Phase	Period	Description	Identification by	Key point
		implementation of automation processes	IT systems	
Industry 4.0	2011–Present	The integration of connected devices, data analytics, and computerized equipment programs has facilitated the automation of industrial production processes	Globalization Digitalization IoT, robotics, big data, cloud computing	Cyber-physical-systems
Industry 5.0	Future	The enhancement of products and services through the collaborative efforts of human intellect and machines	Personalization Robotics and AI Sustainability	Human-robot co-working Bio-economy

2.1 Key Enablers of Industry 5.0

Industry 5.0 is a theoretical framework that expands upon the principles of Industry 4.0, placing emphasis on the incorporation of human-centered methodologies with modern technologies in order to establish industrial systems that are more collaborative and sustainable. Industry 4.0 primarily centers around the integration of automation, data interchange, and intelligent systems, whereas Industry 5.0 places greater significance on the involvement of human agents within the manufacturing domain. Figure 2 deals with Key Enablers of Industry 5.0.

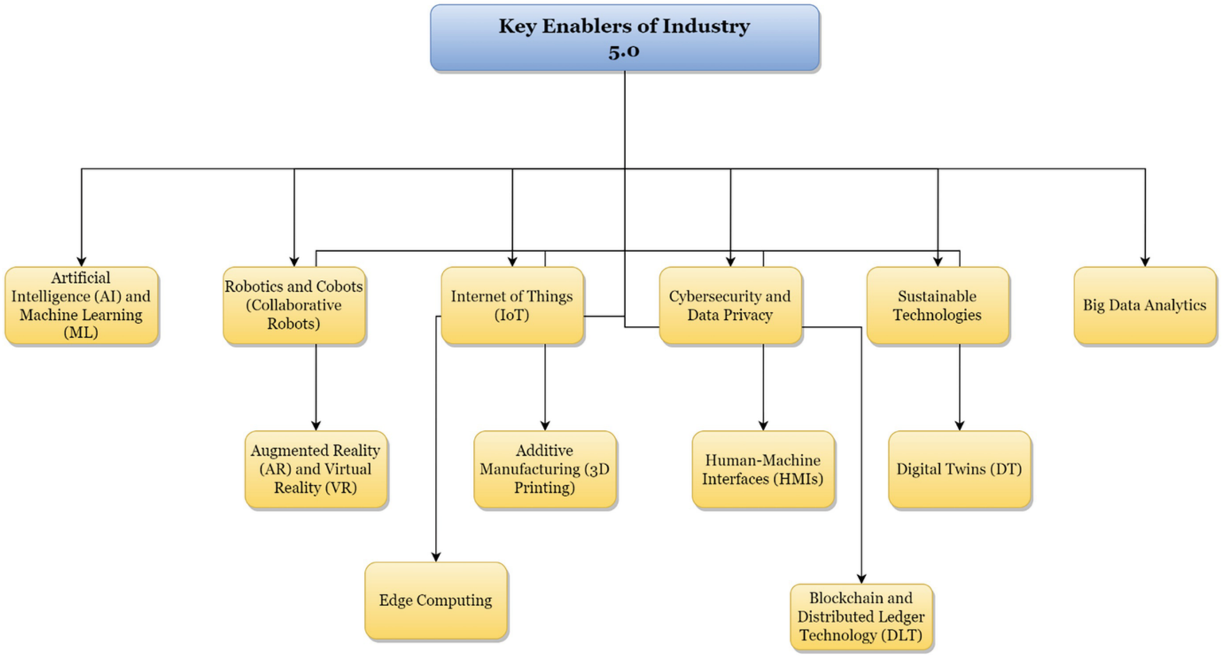

Key Enablers of Industry 5.0
Artificial Intelligence (AI) and Machine Learning (ML)
Robotics and Cobots (Collaborative Robots)
Internet of Things (IoT)
Cybersecurity and Data Privacy
Sustainable Technologies
Big Data Analytics
Augmented Reality (AR) and Virtual Reality (VR)
Additive Manufacturing (3D Printing)
Human-Machine Interfaces (HMIs)
Digital Twins (DT)
Edge Computing
Blockchain and Distributed Ledger Technology (DLT)

Fig. 2 Key enablers of Industry 5.0

The essential technological foundations for Industry 5.0 encompass the following:

1. *Artificial Intelligence (AI) and Machine Learning (ML):* AI and ML are two closely related fields that have gained significant attention in recent years. AI refers to the development of computer systems that can do tasks that typically require human intelligence, such as speech recognition, decision-making, and problem-solving. On the other hand, ML is a subset of AI that focuses on the development of algorithms. These technologies are of paramount importance in the context of Industry 5.0 as they facilitate the ability of machines to acquire knowledge from data and autonomously make decisions without the need for explicit programming. This facilitates the implementation of manufacturing processes that are characterized by increased flexibility and adaptability, enabling them to effectively respond to dynamic human inputs.

2. *Robotics and Cobots (Collaborative Robots):* The topic of discussion pertains to the field of robotics and the emergence of collaborative robots, also known as cobots. Collaborative robots, also known as cobots, are specifically engineered to operate in close proximity to human workers, thereby minimizing the necessity for extensive safety enclosures or barriers. This promotes a heightened level of integration and interactivity inside the workplace, enabling humans and robots to engage in collaborative efforts.

3. *Augmented Reality (AR) and Virtual Reality (VR):* AR and VR are technological advancements that facilitate the superimposition of digital information onto the physical world or the creation of immersive virtual worlds. In the context of Industry 5.0, these technologies have the potential to be employed in various capacities such as delivering immediate instructions to workers, modeling production processes, and facilitating remote cooperation.

4. *Internet of Things (IoT):* IoT refers to the integration of physical items, machines, and sensors with the internet, enabling them to gather and exchange data. This technology facilitates the continuous

monitoring and management of industrial processes in real time, resulting in enhanced operational efficiency and responsiveness.

5. *Additive Manufacturing (3D Printing):* Additive manufacturing, also known as 3D printing, is a manufacturing process that involves the creation of three-dimensional objects by adding material layer by layer. The utilization of 3D printing technology enables the fabrication of intricate and personalized components, frequently in a timely manner as per specific requirements. This technological advancement has the potential to mitigate waste, enhance design adaptability, and provide a greater degree of localized production.

6. *Edge Computing:* Edge computing refers to the activity of doing data processing in close proximity to its source, hence minimizing latency and facilitating expedited decision-making. Real-time control and process optimization are of utmost importance in the context of Industry 5.0.

7. *Cybersecurity and Data Privacy:* The topic of discussion pertains to the field of cybersecurity and the protection of data privacy. Ensuring the security of data and systems is of paramount importance, as is the case with any sophisticated technology. The implementation of robust cybersecurity measures is necessary in order to mitigate cyber attacks and ensure the protection of sensitive information.

8. *Human-Machine Interfaces (HMIs):* HMI refers to the systems that provide interaction between humans and machines. In the context of Industry 5.0, the presence of intuitive and user-friendly interfaces that facilitate human-machine interaction is of utmost importance. These encompass touchscreens, voice recognition, gesture controls, and various other interfaces that enhance the ease of communication between humans and machines.

9. *Blockchain and Distributed Ledger Technology (DLT):* These technologies have emerged as innovative and transformative technologies in recent years. These technologies have gained significant attention and are being widely explored and implemented across several industries. The utilization of these technologies has the potential to augment confidence, transparency, and traceability inside

supply chains and transactions, thereby addressing crucial elements of Industry 5.0’s focus on collaboration and sustainability.

10. *Sustainable Technologies:* Industry 5.0 prioritizes sustainability and the adoption of environmentally-friendly practices. This encompasses many technological advancements aimed at enhancing energy efficiency, minimizing waste generation, and optimizing resource use.

11. *Digital Twins (DT):* A DT refers to the virtual representation of a real system or thing. Real-world entities, such as wind farms, industries, jet engines, buildings, and even more extensive systems like smart cities, have the capability to be digitally represented, as indicated by DT. Despite being initially proposed in 2002, the realization of the notion of digital transformation (DT) has only materialized in recent years, mostly attributed to the rapid proliferation of the Internet of Things (IoT). The implementation of the Internet of Things (IoT) has resulted in the increased cost-effectiveness of digital transformation (DT), thereby expanding its accessibility and affordability across many industries. The data generated by physical items is transmitted to their digital counterparts for the purpose of simulation, facilitated by Internet of Things (IoT) devices. The utilization of digital transformation (DT) enables the mapping of real-time objects and systems, facilitating the analysis and monitoring of their digital counterparts. This capability allows for the anticipation and prevention of issues before they manifest in the physical realm. The swift progress of artificial intelligence (AI), machine learning (ML), and big data analytics has facilitated the implementation of digital transformation (DT) in order to decrease maintenance expenses and enhance system performance.

12. *Big Data Analytics:* The field of big data analytics encompasses the systematic extraction of significant insights and patterns from extensive quantities of structured and unstructured data, which is frequently distinguished by its substantial volume, rapid velocity, and diverse variety. Advanced computational approaches, such as machine learning, data mining, and statistical analysis, are utilized to extract important information that can facilitate informed decision-making and confer a competitive advantage across diverse industries. Through

the utilization of big data, organizations have the ability to discover latent associations, patterns, and anomalies, ultimately resulting in heightened operational efficacy, enriched consumer interactions, and the creation of pioneering offerings.

It is important to note that the adoption of these technologies should be customized to suit the unique requirements and circumstances of individual industries and organizations. Furthermore, it is imperative to evaluate ethical implications and the potential effects on the labor force during the implementation of these technologies in the progression toward Industry 5.0.

2.2 Challenges and Limitations of Industry 5.0

Industry 5.0 effectively addresses the majority of industrial challenges that arise from the displacement of human labor in various processes. However, it is important to consider the integration of supplementary forward-thinking competencies, as there is a possibility that individuals may acquire novel manufacturing proficiencies in the foreseeable future. There are a multitude of abilities that are currently in the process of development, a selection of which will be discussed in this part.

I. In order to integrate advanced talents into industrial management, it is imperative to consider the incorporation of ethical concepts inside autonomous systems.

II. There exists a necessity to ensure the appropriate verification and validation of ethical conduct within the autonomous system framework.

III. The significance of implementing operational transparency and achieving rapid and competent manufacturing can be observed in the context of an overproduction situation.

IV. The results of the study should yield comprehensible answers for ethical behavior within an independent framework. Industrial experts are currently encountering challenges related to the adaptation and execution of their practices.

V. Tuning and validation processes are crucial in mitigating potentially significant issues that may arise among various stakeholders, including technological specialists, stockholders, society, and enterprises.

2.3 Edge AI and Its Transition into Industry 5.0

In the contemporary era of the IoT and edge, there is a significant volume of data generated by interconnected devices that requires collection and analysis. Consequently, a substantial quantity of data is generated instantaneously, thereby requiring the utilization of artificial intelligence (AI) systems to comprehend and analyze its magnitude. The term “edge AI” pertains to the deployment of artificial intelligence (AI) software on physical computing devices situated in various real-world settings. Every industry within the business sector is actively pursuing the use of automation technologies in order to optimize workflow processes, increase production levels, and strengthen overall security measures. The utilization of advancements in edge AI has enabled robots and gadgets to operate with a level of cognitive “intelligence” comparable to that of human beings, regardless of the specific context in which they are deployed. The fundamental emphasis of Industry 5.0 is on the development of an industry that is resilient, human-centric, and sustainable, serving as a complementary aspect to Industry 4.0. Inquiry pertains to the orchestration of the transition into Industry 5.0, wherein the focal point lies in the interaction between humans and machines. The emergence of Industry 5.0 stems from the recognition that the forthcoming era of manufacturing must transcend its conventional concentration on consumer-centric production, as consumer demands consistently gravitate toward swifter, superior, and more cost-effective outcomes. 5.0 is utilizing technology to establish a corporate entity that prioritizes the interests of both consumers and workers, while also emphasizing sustainability and a strong commitment to the local operational environment. According to Gartner’s Magic Quadrant, edge AI is now positioned at the apex of exaggerated expectations, indicating its lack of maturity at present.

Industry 5.0 signifies the subsequent advancement in manufacturing and industry, expanding upon the groundwork established by Industry 4.0. The convergence of IoT, edge computing, and AI (ML) is at the vanguard of

Industry 5.0, initiating a paradigm shift for several businesses. Internet of Things (IoT) devices produce large quantities of live data, which is analyzed at the edge to minimize delays and improve speed of response. Artificial intelligence, namely machine learning, examines this data to extract practical insights and enhance procedures. The collaboration between different components in Industry 5.0 allows for the attainment of unparalleled levels of automation, efficiency, and customization. The integration of IoT, edge computing, and AI enables enterprises to make intelligent decisions, perform predictive maintenance, and implement adaptive production processes. This convergence not only improves operational efficiency and productivity, but also promotes a more adaptable and responsive industrial ecology, where machines and humans collaborate together to stimulate innovation and achieve sustainable growth. The potential impact of this combination on the industry is as follows:

Real-Time Decision-Making

- IoT Sensors: Internet of Things (IoT) sensors installed on the shop floor gather real-time data from machines and equipment.
- Edge Computing: Edge devices perform local data processing, resulting in reduced latency and facilitating rapid decision-making at the point of origin.
- AI/ML: Artificial intelligence and machine learning algorithms examine patterns and irregularities in data that is constantly updated, offering valuable insights for prompt decision-making.

Predictive Maintenance

- IoT Sensors: Sensors are utilized to continuously monitor the health and performance of equipment.
- Edge Computing: Edge devices perform data analysis to identify initial indications of equipment malfunction.
- AI/ML: Artificial intelligence and machine learning algorithms are utilized to forecast maintenance requirements, hence minimizing operational interruptions and preempting unforeseen malfunctions.

Optimal Resource Allocation

- Internet of Things (IoT) Devices: Sensors monitor and measure the utilization of resources, such as electricity and raw materials.
- Edge Computing: The practice of doing data processing and analysis at the edge of a network, close to where the data is generated, in order to efficiently utilize resources according to immediate need.
- AI/ML: Artificial intelligence and machine learning algorithms enhance processes by optimizing them, resulting in waste reduction and overall efficiency improvement.

Supply Chain Optimization

- Internet of Things (IoT) Devices: Sensors deployed throughout the supply chain monitor and record the flow of commodities.
- Edge Computing: Edge devices locally process supply chain data to ensure rapid reaction.
- AI/ML: Artificial intelligence and machine learning algorithms improve supply chain logistics by forecasting demand, minimizing delays, and increasing overall efficiency.

Customization and Personalization

- IoT Devices: Gather data regarding client preferences and patterns of product utilization.
- Edge Computing: Locally process this data to facilitate rapid customization.
- AI/ML: Machine learning models offer tailored product recommendations and improve customization capabilities.

Improved Safety and Quality Assurance

- Internet of Things (IoT) Sensors: Continuously monitor environmental conditions and safety factors.
- Edge Computing: Perform local data analysis to guarantee prompt answers to safety problems.
- AI/ML: Machine-learning algorithms detect patterns that could potentially signify quality problems, thereby enhancing product quality and safety.

Data Privacy and Security

- Edge Computing: The utilization of local processing minimizes the necessity of transmitting sensitive data to centralized servers.
- AI/ML: Sophisticated security algorithms can be implemented at the edge to safeguard data and systems.

Human-Machine Communication

- Internet of Things (IoT) Devices: Establish connections between workers and machines to facilitate smooth communication.
- Edge Computing: Local processing enables efficient and rapid communication between humans and machines.
- AI/ML: Machine learning optimizes interactions between humans and machines, enhancing cooperation and the process of making decisions.

To summarize, the incorporation of IoT, edge computing, and AI/ML in Industry 5.0 results in a fundamental change toward smarter, more reactive, and more effective industrial procedures, promoting a more interconnected and optimized environment. This convergence facilitates the ability of industries to adjust to evolving demands, decrease expenses, and enhance overall output.

---

3 Existing Works: An Extensive Survey

An extensive literature study on the convergence of edge computing and artificial intelligence/machine learning (AI/ML) involves a meticulous examination of the current knowledge base, including influential research, modern studies, and emerging patterns in both fields. The evaluation should outline the core principles of edge computing, clarifying its structure, essential elements, and the transformative change it brings by processing data in proximity to its origin instead of relying on centralized cloud infrastructures. It should explore the ever-changing field of AI/ML, covering fundamental theories, techniques, and applications in various industries. The literature synthesis should clarify the connections and difficulties involved in combining edge computing with AI/ML. It should emphasize how being close to data sources improves real-time processing, decreases latency, and boosts the overall effectiveness of AI algorithms. Furthermore, the review should thoroughly examine the consequences of this convergence on many industries, including healthcare, manufacturing, and IoT, elucidating the transformative capacity and the related ethical considerations. The literature review should offer a thorough comprehension of the current status, future possibilities, and potential areas for additional investigation at the intersection of edge computing and AI/ML by carefully examining existing research, detecting gaps, and recognizing emerging patterns.

An exhaustive literature analysis on the convergence of edge computing and AI/ML should offer a thorough synopsis of current research, advancements, and patterns in both domains. The following are the essential prerequisites for a literature review on edge computing and AI/ML for prospective researchers:

Fundamental Ideas

- Provide concise definitions and explanations for essential topics in edge computing, such as edge devices, edge nodes, and edge analytics. Additionally, clarify key concepts in AI/ML, including machine learning techniques and deep learning.
- Elucidate any specialized vocabulary or conceptual frameworks that are crucial for comprehending the text.

Obstacles in Integration

- Examine the difficulties and concerns associated with the fusion of artificial intelligence/machine learning with edge computing.
- Examine the impact of latency, bandwidth, and resource limitations on the integration process.

Architectures and Frameworks

- Examine several architectural methodologies and frameworks for deploying artificial intelligence and machine learning on the edge.
- Examine the standardization initiatives and prevalent architectural frameworks discussed in the literature.

Assessment of Performance

- Analyze research that evaluates the efficacy of artificial intelligence and machine learning models in edge computing scenarios.

- Incorporate deliberations on measurements, standards, and procedures employed for evaluation.

Issues Pertaining to the Protection of Data and Personal Information

- Examine scholarly works that discuss the security and privacy difficulties that arise when implementing artificial intelligence and machine learning at the edge.
- Examine the possible remedies or measures suggested in the literature.

Areas of Research That Have Not Been Adequately Explored or Understood

- Identify deficiencies or domains where additional investigation is required.
- Examine the obstacles that have not been sufficiently tackled in the current body of literature.

Potential Avenues for Future Development

- Offer analysis on prospective advancements in the fusion of edge computing and AI/ML.
- Examine current and developing patterns, advancements, or areas of study.

Use Cases

- Provide a concise overview of practical situations and scenarios in which the combination of edge computing and AI/ML has been effectively implemented.
- Discover several sectors including health care, manufacturing, IoT, and further fields.

Xiong et al. [1] implemented a resource allocation policy to fulfill their objective (to reduce the weighted sum calculated from the average number of requested resource and time of job completion) to boost the resource utilization efficiency of edge computing in Internet of Things. In MEC (Mobile Edge Computing), they used MDP (Markov decision process) for formulating the problem of allocating resources and applied DRL (deep reinforcement learning) approach to solve the problem. And they proposed an improved version of deep Q-network (DQN) algorithm, where numerous replay memories are applied to collect tiny mutual influences individually. Their enhanced version of the DQN algorithm exhibits superior performance compared to the original version, as evidenced by simulation outcomes. However, the adjustment in action selection inadvertently impacts the average completion time per task, as detailed in their publication. Their policy results in an average time of 4:2, which surpasses the average times of the reference policies, recorded at 3:53 and 3:52 respectively (as per their article). This is primarily attributed to insufficient resources allocated for requested jobs, leading to additional processing time as queued jobs are scheduled across various time slots.

Y. Zhang et al. [2] explored a Het-MEC (heterogeneous multi-cloud mobile edge computing) environment in their study. Here, individual cloud servers distribute their computing tasks among Mobile Edge Computing (MEC) servers and Edge devices (EDs), competing for both transmission and computing resources. They proposed a distributed scheme for each CC (cloud center) to determine its resource allocation and data offloading policies individually to reduce the latency of system. They designed MARL (multi-agent reinforcement learning)-based algorithm to represent the competition between multiple cloud centers where every cloud center is self-motivated to modify their behavior by acquiring a knowledge of explicit models of another cloud centers.

Abdullah Khan et al. [3] described in their paper on the use of social Internet of Things concept and machine-learning algorithm to make easier the process of allocating edge resources to reduce the edge computer allocation perplexity. To achieve trustworthy and proper selection of edge computers to computational task, they divided the process into two parts. The first one is the trustworthy community detection phase (TCDP), where they divided the IoT devices into three different categories (co-location−/co-work-based relation (CLOR), social friendship and ownership relation (SFOR), and social object relationship (SOR)) according to their relation, geographical location etc. The community detection was done by Louvain algorithm. The second one is the matching device phase (MDP), whose main motive is to allocate the edge computer and predict the time needed by edge computer to process the requested task for same community. Their proposed framework worked well only for devices that belonged to the same categories; devices that do not belong to the same category were not studied in depth. In this approach, the computing process depends on historical data. Security and privacy of data collected from different users were not addressed.

X. Liu et al. [4] explored the challenges associated with implementing a task offloading policy, particularly due to the scarcity of transmission spectrum resources required for IoT users to offload computation tasks to Edge servers. In response to this challenge, the authors proposed a Task Offloading policy leveraging Machine Learning algorithms, introducing a two-step implementation approach. The first is the clustering of IoT users based on the priority of the IoT users, i.e., higher and lower priority. Lower priority is assigned as local computing and higher priority as edge computing. K-means-based clustering algorithm was used to cluster the IoT users. In the second step, they employ distributed task offloading, utilizing MDP (Markov Decision Process) as the model for determining an optimal offloading policy for computation. To learn this offloading policy, they introduce a DQN-based computing algorithm. Simulation results demonstrated that the scheme grouped the user successfully and the system cost was lower than the other baseline schemes. This approach is cost-effective compared to the other three baseline schemes, but it has energy consumption and task execution latency issues; moreover, this approach is only for specific users and not for multiple users. Training time for DQN learning will increase proportionally with the number of episodes, which increases the complexity. Also, the system cost will increase with the increase in computation task.

According to Zhu Tianqing et al. [5], allocating computing resources in IoT-edge computing is a significant issue. Most of the previous approaches focused on either central or edge server, both of which have their own drawbacks. Central servers are capable of allocating optimal resources compromising privacy, whereas edge servers are not much capable of allocating computing resource. The authors proposed a novel resource allocation policy without compromising privacy as concurrent federated reinforcement learning and used DRL (deep reinforcement learning) or allocating computing resources. Their simulation results demonstrate that their approach effectively reduces task execution time, requires less communication overhead, and achieves a better utilization rate of resource allocation. However, during the initialization stage, the edge host requires more time compared to other approaches. This is due to multiple parties learning simultaneously and jointly, where a negative impact on one party can affect others, leading to increased time consumption. Additionally, this method lacks detailed elaboration on privacy concerns.

According to Wenlong Guo et al. [6], blockchain can be applied to various Internet of Things (IoT) platforms. The efficiency of the blockchain depends on how to allocate computational resources, participation of devices, and also on its computing capability. They discussed an edge-computing network based on blockchain, where the edge service provider (ESP) provides computational resources for the minors (the users or edge node participating in the computing process). They investigated an efficient incentive mechanism for the minors to purchase computational resources. According to the authors, a two-stage Stackelberg game is formulated between ESP and the minors. They explored the Stackelberg equilibrium of the optimal mining strategy under two different mining schemes. Their aim was to find the optimal incentive for minors and the ESP to choose auto-fit strategies, and their theoretical analysis and numerical simulation demonstrated that the proposed scheme was very effective for encouraging devices to participate in the blockchain. All seems to be in order, but the authors assume that no minors (the users or edge nodes participating in the computing process) possess any hash power; instead, all hash power is owned by the ESP (Edge Service Provider). Additionally, minors are required to purchase hash power from the ESP, potentially posing difficulties due to high costs.

Xiaolan Liu et al. [7] investigated that edge users in different IoT edge computing applications (smart wearable, virtual reality, mobile games) are facing difficulties to offload the computation and allocate resource due to their repeated RAT (radio access technology) request and their huge requirements of computation and spectrum resources. To overcome these issues in IoT edge computing, the authors proposed a scheme for computation offloading along with resource allocation and formulated as stochastic game. To solve stochastic game, they developed multiagent reinforcement learning framework with multi-agent Q-learning (IL-based MA-Q) algorithm that is capable of independent learning. Stochastic game has the ability to learn and act as a learning agent that help minimize the system cost. Their simulations demonstrate the successful establishment of a computation offloading and resource allocation policy, resulting in a reduction in the overall system cost and a decrease in the computational burden on the gateway. Moreover, the scheme will generate worse average cost if the user explores too many random actions instead of exploiting the optimal action. Also with the minimal growth of end users, the average cost increases.

Yuqi Fan et al. [8] introduced a blockchain-based resource allocation and computation offloading function for IoT-edge computing. While blockchain has the drawback of consuming high computing resources and power, it presents a better way to offload computational tasks from limited-capability IoT devices to Cloud Edge Service Providers (CESP). CESP is responsible for selecting the server from edge or cloud to compute the offloaded task, aiming to boost service provisioning and reduce costs. They formulated the computing offloading framework as a Stackelberg Game, where CESP acts as the commander and users (Minors) act as assistants. To achieve their objective of increasing revenue for CESP and users, they proposed an iterative greedy-and-search (IGS)-based algorithm to solve pricing at CESP and the resource allocation problem, which is modeled as a mixed-integer programming problem (MIP). This algorithm addresses two subproblems: resource allocation within a specific price range and pricing the resources of a particular scheme of allocating resources, both of which contribute to revenue optimization for CESP. Simulation results demonstrated that their proposed IGS algorithm helps increase revenue for both CESP and users. While the authors primarily focused on revenue growth, they briefly mentioned that this algorithm could also reduce delay, although they did not elaborate on this aspect. Additionally, they did not extensively discuss security, suggesting that this scheme may compromise security.

Xianwei Li et al. [9] discussed various services and applications (e.g., autonomous vehicles, smart health care) benefiting from IoT and IIoT advancements. However, due to the limited computing capability of IoT devices, task offloading to cloud or edge servers is necessary, albeit with some challenges such as network traffic and latency. To address these challenges, they studied multi-user-based mobile edge computing (MEC), which cooperatively allocates communication and computing resources to reduce energy consumption and latency. They formulated the problem as a mixed integer nonlinear programming (MINLP) problem, dividing it into two subproblems: optimal local computing and optimal edge computing. They proved that the problem is NP-hard. Simulation results showed that their proposed model could reduce energy consumption, latency, and improve offloading decisions, though they did not discuss big data management.

Ying He et al. [10] highlighted the rapid growth of IoT devices, posing challenges for centralized/cloud computing systems due to requirements for minimal latency and greater bandwidth. They advocated for Mist/Fog computing (edge-centric computing) to overcome these challenges. They discussed security and privacy aspects of IoT-edge computing, important characteristics of blockchain for edge computing, and proposed a blockchain-based edge computing system for IoT. They described a contract between private blockchain networks leveraging Asynchronous Advantage Actor-Critic (A3C), demonstrating how blockchain and artificial intelligence (AI) can work jointly. Simulation results indicated that the resource allocation scheme was effective, although they did not address transmission delay or fully resolve task drop issues. Their scheme directly depends on computational processing speed, suggesting that it can meet requirements if processing speed is sufficiently high.

Xu, Jianqiang et al. [11] noted the rapid growth of IoT devices and the offloading of intensive tasks to Mobile Edge Centers (MECs) for computation. However, they identified a load balance problem due to limited resources and coverage of edge devices. To address this, they proposed a computing offloading and resource allocation algorithm using game theory. They modeled the architecture for multi-user and multi-MEC scenarios and developed a prediction method to estimate the waiting time for offloading tasks to MEC servers. Energy consumption and latency issues were treated as a multi-user game problem. They concluded that Nash equilibrium always exists in this problem and can be solved by an optimal computing and resource allocation method. Simulation results indicated lower task execution costs, but performance degradation with increasing users, highlighting a lack of support for communication switching for mobile users.

Deng et al. [12] emphasized the vital role of edge computing (EC) in managing a large number of IoT devices efficiently. They acknowledged that EC could provide effective services but noted challenges related to trustworthiness and adherence to service-level agreements (SLAs). To address these, they proposed a trained resource allocation scheme using reinforcement learning. Their experiments showed improved trustworthiness and performance, although they noted drawbacks such as task redirection to cloud servers with increasing workload, leading to increased latency.

Alqerm et al. [13] discussed the exponential growth of IoT applications and the inadequacy of existing computing models to handle delay-sensitive and high-bandwidth applications. They proposed an enhanced centralized online Q-learning-based resource allocation scheme to maximize the utility of IoT applications and ensure fair resource allocation. Numerical results demonstrated improved application utilities and allocation fairness, but they cautioned about the computational complexity of centralized online Q-learning increasing with the growth of IoT devices.

Woongsoo et al. [14] explained that most IoT systems operate in automated environments, generating large amounts of real-time data. To reduce the load on data centers, cloud computing is replaced by edge computing. They described an edge-based IoT system where edge gateways (EGs) preprocess data from end devices (IDs) to reduce the load on edge servers (ESs). However, due to limited computing resources, EGs cannot process all data, necessitating direct connections between IDs and ESs. They proposed a Lagrangian and Karush–Kuhn–Tucker (KKT)-based frequency resource allocation scheme to manage interfaces and allocate resources according to EGs' and ESs' computing capacity. Performance evaluation showed improvements in data reception rate, latency, and workload fairness, though there were occurrences of loss due to lack of computing capacity in some cases.

Chen et al. [15] discussed the increasing mobile traffic due to the proliferation of intelligent mobile devices, posing challenges for fifth-generation (5G) communication. They proposed using artificial intelligence (AI) to control data traffic, developing a long short-term memory (LSTM)-based deep learning algorithm to predict traffic flow peaks. Their scheme effectively reduced packet loss and latency, although they noted open issues such as users' mobility prediction and risk perception in edge-cloud sharing.

Atapattu et al. [16] defined the Internet of Things (IoT) with mobile edge computing (MEC), where limited-resource capable IoT devices offload computational tasks to edge servers located nearby. They proposed a framework for jointly offloading sequential tasks and allocating computing and communication resources. Their optimization methods aimed to limit task processing delay, reduce energy consumption, and jointly optimize task offloading strategies using fast fading and slow fading channels. Simulation results showed effectiveness compared to previous methods, though they did not extensively discuss security and privacy concerns.

To overcome the challenge, they break the problem into a decision of task offloading problem for one dimensional search and task offloading decision with non convex optimization problem. Non-convex problem is converted into convex problem and is solved by golden search method. The stochastic optimization problem in a fast fading channel which optimizes the instant decision of offloading the next subtask, considering the task offloading delay. Based on state of instant channel, optimal online policy is derived in fast-fading channel. The results showed that their approach to be comparatively effective than the previous methods. If the computing capabilities of slow-fading channels are not high, then the consumption of time for computing tasks will increase. However, the authors didn’t discuss about security and privacy for resource allocation.

C. Liang et al. [17] developed an alternating direction method of multipliers (ADMM)-based algorithm to solve resource allocation, computation offloading, and content caching strategy problems arising from network traffic growth in mobile edge computing for wireless cellular networks. They transformed the problems into convex ones and utilized MATLAB simulation to demonstrate a decrease in loading percentage with an increase in small cell deployment.

Zhenquan et al. [18] presented a workload resource allocation scheme using an application prediction (AP) algorithm to reduce response delay. They identified workload allocation as a challenge due to limited computing and storage resources at edge servers in IoT networks. They formulated the problem as a two-part task, involving application prediction and workload allocation. Their simulation results showed that their approach had lower response times compared to other strategies, but average completion time increased with access device count.

B. Palanisamy et al. [19] discussed latency issues in various IoT applications, suggesting optimal computing resource allocation as a solution. They proposed a decoupled resource allocation scheme called Zeinth, allowing service providers to share resources based on contracts. Their experiments showed better response time and resource utilization, though they noted the potential for higher utilization with enhanced micro data centers (MDCs).

L. Zhao et al. [20] discussed the challenges posed by IoT-based smart cities, with a large number of heterogeneous IoT devices requiring optimal resource allocation for satisfactory performance. They advocated for deploying storage and computing resources near IoT devices and running applications on distributed edge servers. Their study focused on minimizing average service response time through edge resource allocation.

M. K. Ehsan et al. [21] highlighted the exponential growth of different IoT devices and their inability to handle applications requiring extensive computation with minimal delay. They proposed a mobile edge cloud (MEC) solution for offloading computation, aiming to fulfill the requirements of fifth generation communication systems. Their convex optimization problem aimed to minimize task duration by allocating local and edge computational capabilities and optimizing task segmentation. Simulation results demonstrated fundamental trade-offs between partial offloading and edge computing, with impacts from device capabilities and task requirements.

J. Ren et al. [22] discussed delay reduction in time-division multi-user multiple access MECO systems by jointly allocating computation and communication resources. They proposed local compression, partial compression offloading, and edge cloud compression computing architectures, building expressions to reduce system latency and allocate resources efficiently. Their optimal algorithm reduced system latency, with partial compression offloading exhibiting the lowest latency.

H. Liao et al. [23] emphasized the potential of edge computing in supporting Industrial Internet of Things (IIoT) implementation, with the caveat of performance compromises due to limited resources and contextual unawareness. They proposed a learning-based framework for channel selection, aiming to maximize throughput while respecting energy and reliability constraints. Their simulations demonstrated reliability and effectiveness under multi-MTD and single-MTD scenarios.

T. X. Tran et al. [24] discussed a wireless multi-cell network with MEC servers aiding mobile users in executing intensive computational tasks. They studied joint task offloading and resource allocation, formulating the problem as a mixed integer nonlinear program (MINLP). Simulation results showed effectiveness in optimizing task offloading decisions and resource allocation, though challenges remain in finding optimal solutions for large-scale networks.

P. Wang et al. [25] proposed the EdgeFlow system for multilayer data flow processing, utilizing computing capability across the network layers. They categorized edge devices (EDs), mobile edge computing (MEC), and cloud center (CC) into bottom, middle, and top layers, respectively. Their resource transmission scheme optimized task allocation to multiple layers based on data generation speeds, distinguishing between non-blocking and blocking states. Implementation on Intel Next Units of Computing and universal software radio peripherals demonstrated EdgeFlow’s superiority over previous frameworks.

S. Nath et al. [26] addressed computation offloading and resource allocation in multi-user, multi-channel MEC systems, where each user equipment (UE) requires time-sensitive and high-accuracy computation. They used deep deterministic policy gradient to determine task offloading, wireless medium selection, and MEC resource calculation, aiming to minimize energy consumption, UE deadline penalties, and delay.

In his article, M. Satyanarayanan [27] explores the potential benefits that Edge Computing brings to the Internet of Things, including scalability, privacy-policy enforcement, and the ability to conceal transient cloud disruptions. Additionally, Edge Computing promises highly responsive cloud services for mobile computing.

In their paper, S. Pal et al. [48–54] elucidated various topics pertaining to the convergence of artificial intelligence and the Internet of Things (IoT) for Industry 5.0, including an exploration of an artificial intelligence-based approach to IoT applications in the industrial domain. Additionally, they present an intelligent task scheduling model tailored for the hybrid IoT and cloud environment, specifically addressing the challenges of managing big data applications efficiently. Furthermore, the paper delves into dynamic data optimization techniques within IoT-assisted sensor networks, focusing on their integration with cloud platforms to enhance scalability and performance. Finally, the authors propose an optimized energy efficient strategy aimed at reducing data transmission between edge devices and the cloud within the cloud-IoT ecosystem, emphasizing the importance of resource optimization and sustainability in IoT deployments. Table 2 deals with the survey table of the literature review.

Table 2 Survey table

Sl no	Algorithm used	Utility of the algorithm	Platform/simulator/mathematical model used	Limitations
1	DRL (deep reinforcement learning) [1]	To allocate the computing resources To improve DQN for reducing the Loss than the Original DQN During long run, it asks for less computing request from MEC	1. Resource allocation, System latency problem formulate as MDP (Markov decision Process) 2. Apply DRL (deep reinforcement learning) approach to solve the problem 3. To learn the problem, they used deep Q-network (DQN) algorithm 3.1 To improve DQN, they collected tiny mutual influences by applying numerous replay memories	1. DRL needs huge amount of previous data to study
2	Multi-agent reinforcement learning (MARL)-based algorithm [2]	Resource allocation System latency	1. Multi cloud, distributed computing model	1. As they are using multi-cloud, this will be very expensive
3.	Social Internet of Things concept and machine-learning algorithm [3]	Reduce the complexity of allocation of required edge computer	1. Community detection was done by Louvain algorithm 2. Matching device phase (MDP) to predict the required time of edge computer	1. Reduce allocation perplexity for only same category of relation, not for different relation
4.	Machine-learning algorithm [3]	Reduce the system cost by offloading task	1. K-means-based clustering algorithm to cluster the IoT users 2. MDP (Markov Decision Process) as model of the problem of making offloading policy for optimal computation 3. DQN-based computing algorithm to learn the policy	Training time for DQN learning will increase proportionally with the number of episodes which increases the complexity
5	Concurrent federated reinforcement learning [5]	To achieve global-level optimality for resource allocation without compromising privacy	1. Federated learning for privacy 2. Deep reinforcement learning for resource allocation 3. Python simulator via PyTorch 1.2.0	1. At the initializing stage, edge host needs more time than other approach 2. As multiple parties learn simultaneously and jointly, a negative party can affect other parties, which increases the

Sl no	Algorithm used	Utility of the algorithm	Platform/simulator/mathematical model used	Limitations
				consumption of time
6	Blockchain [6]	Decentralized computing structure can be formed To improve security and privacy of computing	1. Stackelberg game theory to build the optimal mechanism 2. Two-stage iterative algorithm is used to reach SE (Stackelberg equilibrium)	The minors have to buy the hash power from ESP, which is difficult because of high cost
7	Multiagent reinforcement learning [7]	For resource allocation and computation offloading to reduce average cost	1. Stochastic game is formulated from the problem of resource allocation and computation offloading 2. IL-based MA-Q computation offloading algorithm for solving the problem (offloading computing task and allocating of resources) which is formulated as stochastic game	The scheme will generate worse average cost if the user explores too many random actions instead of exploiting the optimal action. Also with the minimal growth of end users, the average cost increases
8	IGS (iterative greedy-and-search)-based algorithm [8]	Increase the revenue of CESP and user	1. Formulate computing offloading framework as Stackelberg game 2. Pricing at CESP and resource allocation problem is modeled as mixed-integer programming problem (MIP)	Less secure, not optimized for task completion delay, requirements of computation resource is high
9	Edge computing resource allocation algorithm based on game theory [11]	Resource allocation Computing task offloading Fixed load balance problem occurs during computing task offloading and allocating	Design multiuser, MEC computing offloading, and resource allocation model A prediction mode to predict waiting time of task for offload to MEC However, energy consumption and latency problem are formulated as multiuser game problem	Overall system performance is inversely proportional to the number of user The scheme does not support communication switching for mobile users

---

4 AI-Based Trustworthiness in Edge Systems

Within the framework of Industry 5.0, it becomes crucial to prioritize the establishment of reliability in AI-driven systems implemented at the edge. In order to accomplish this, a series of stringent criteria must be fulfilled. First and foremost, it is crucial to implement strong security measures that include end-to-end encryption, secure boot processes, and intrusion detection in order to protect against possible cyber threats. Ensuring reliability is of utmost importance, requiring the incorporation of fault-tolerant techniques and redundancy to minimize the possibility of system failures. Transparency and explainability are crucial for establishing trust in AI algorithms, necessitating the creation of interpretable models and systems that can clearly communicate their decision-making processes. Interoperability is crucial, as AI systems deployed at the edge need to smoothly interface with established industrial infrastructures and adhere to existing standards. Additionally, it is crucial to engage in ongoing surveillance and make regular modifications in order to adjust to changing risks and uphold peak performance. Ultimately, it is crucial to integrate ethical concerns into the design of AI, guaranteeing compliance with ethical principles, rules regarding data protection, and responsible practices in AI. This is necessary to promote a reliable and socially accountable deployment of AI in the ever-changing context of Industry 5.0.

The concept of AI-based trustworthiness in edge systems pertains to the dependability, confidentiality, and ethical conduct of artificial intelligence algorithms and models that are implemented on edge computing devices. The concept of edge computing entails the localization of data processing in close proximity to its source, resulting in diminished latency and bandwidth consumption in contrast to the transmission of data to a centralized cloud server. The establishment of trustworthiness is of utmost importance in order to guarantee the secure, dependable, and ethical functioning of artificial intelligence models that are implemented on edge devices. The following are certain factors to be taken into account in order to guarantee the reliability and credibility of artificial intelligence (AI)-powered edge systems as shown in Fig. 3:

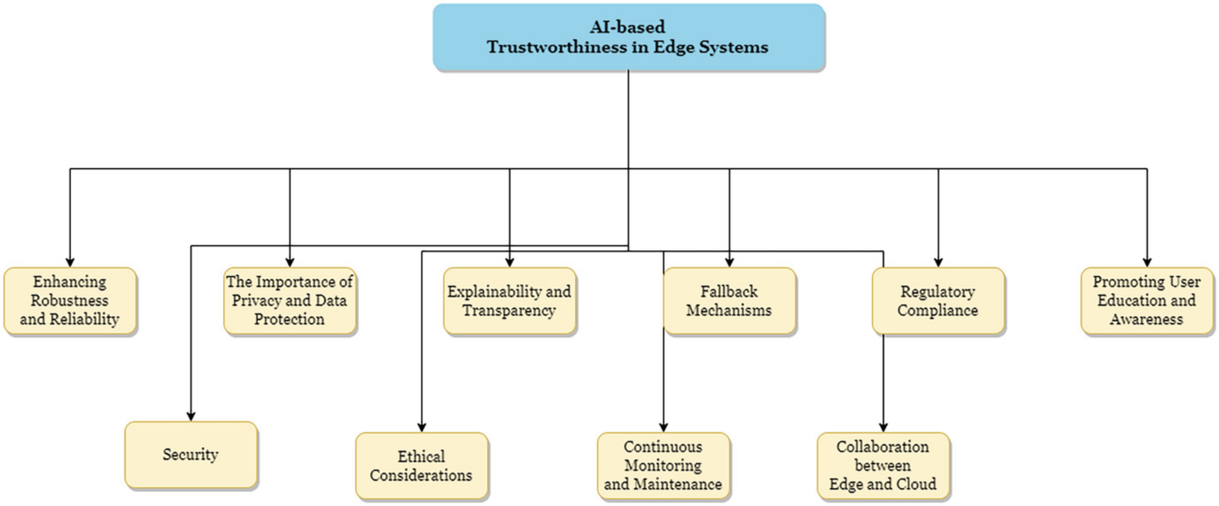

AI-based
Trustworthiness in Edge Systems
Enhancing Robustness and Reliability
The Importance of Privacy and Data Protection
Explainability and Transparency
Fallback Mechanisms
Regulatory Compliance
Promoting User Education and Awareness
Security
Ethical Considerations
Continuous Monitoring and Maintenance
Collaboration between Edge and Cloud

Fig. 3 AI-based trustworthiness in edge systems

1. *Enhancing Robustness and Reliability:* It is imperative for artificial intelligence (AI) models implemented at the edge to exhibit robustness in the face of diverse environmental circumstances and input fluctuations. It is imperative that these systems maintain their operational accuracy despite the presence of noise, outliers, or unanticipated inputs.

2. *Security:* Edge devices frequently possess constrained resources, rendering them susceptible to potential security breaches. In order to safeguard AI models and data stored on these devices, it is imperative to incorporate security measures, including but not limited to encryption, access limits, and routine security upgrades.

3. *The Importance of Privacy and Data Protection:* Edge devices have the capability to handle sensitive or personal data. It is of utmost importance to ensure adherence to privacy standards and establish robust data protection measures. Methods such as differential privacy can effectively safeguard individual privacy while yet enabling the extraction of valuable insights.

4. *Ethical Considerations:* It is imperative to ensure that AI models are developed and trained in a manner that mitigates the presence of bias, discrimination, and other ethical concerns. This entails the examination of the equity of projections among various demographic

cohorts and the prevention of perpetuating detrimental preconceptions.

5. *Explainability and Transparency:* The requirement for AI models deployed at the edge is that they possess explainability and transparency, which refers to the ability of these models to provide understandable and interpretable conclusions for humans. The establishment of trust and adherence to regulatory requirements necessitating the provision of explanations for judgments guided by artificial intelligence are crucial factors.

6. *Continuous Monitoring and Maintenance:* It is imperative to have a continual monitoring process for edge systems in order to assess their performance, security, and adherence to ethical standards. It is imperative to consistently administer regular maintenance, updates, and patches in order to uphold the enduring reliability of the AI models.

7. *Fallback Mechanisms:* In the event of unforeseen problems or faults, it is imperative for edge systems to incorporate contingency measures to guarantee the continued execution of essential functions.

8. *Collaboration Between Edge and Cloud:* Although edge computing facilitates localized data processing, it may necessitate interaction with cloud-based systems to perform tasks such as model updates, consolidating insights, or collaborating with other edge devices. It is advisable to utilize secure communication methods for such interactions.

9. *Regulatory Compliance:* In order to ensure adherence to certain norms and standards, it is necessary to consider the application and location. The maintenance of adherence to these stipulations is crucial in establishing credibility.

10. *Promoting User Education and Awareness:* It is imperative to ensure that users and stakeholders are adequately informed about the capabilities and constraints associated with AI systems deployed on edge devices. Effective communication regarding the utilization of

data and the potential consequences of decisions led by artificial intelligence (AI) has the capacity to foster trust.

In a nutshell, the establishment of trustworthiness in edge systems that utilize artificial intelligence necessitates a comprehensive methodology that encompasses several aspects such as resilience, safeguarding, confidentiality, moral principles, openness, and adherence to legal frameworks. Continuous monitoring and adaptation of systems to accommodate new risks and requirements are of paramount importance.

5 Applications of AI in Edge Computing

The concept of AI in edge computing pertains to the incorporation of AI algorithms and models onto edge devices or within edge servers, facilitating the local processing of data and autonomous decision-making, thereby reducing dependence on centralized cloud resources. There are several advantages associated with this approach, which encompass decreased latency, enhanced privacy, and diminished bandwidth demands. The following are several specific uses of artificial intelligence (AI) within the context of edge computing, presented in points as depicted in Fig. 4.

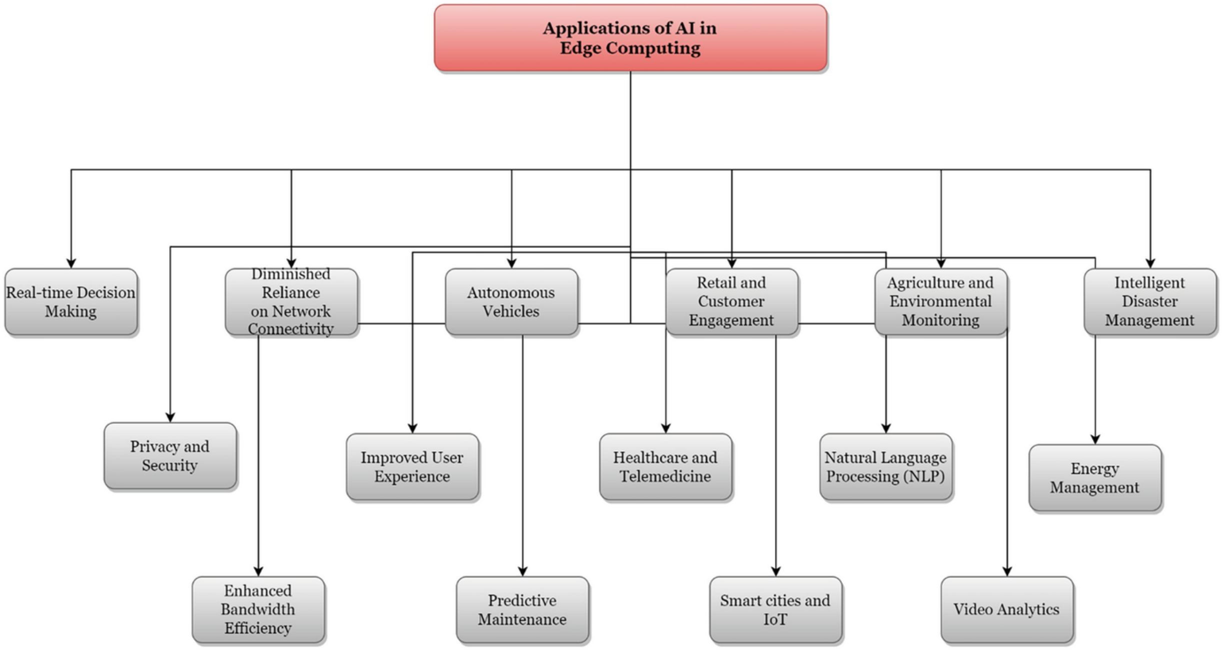

Applications of AI in Edge Computing
Real-time Decision Making
Diminished Reliance on Network Connectivity
Autonomous Vehicles
Retail and Customer Engagement
Agriculture and Environmental Monitoring
Intelligent Disaster Management
Privacy and Security
Improved User Experience
Healthcare and Telemedicine
Natural Language Processing (NLP)
Energy Management
Enhanced Bandwidth Efficiency
Predictive Maintenance
Smart cities and IoT
Video Analytics

Fig. 4 Applications of AI in edge computing

1. *Real-Time Decision-Making:* The utilization of AI at the edge facilitates prompt processing and decision-making, eliminating the necessity of transmitting data to a centralized cloud server. Low latency is of utmost importance in applications such as driverless vehicles, industrial automation, and augmented reality.

2. *Privacy and Security:* Edge computing facilitates the localization of sensitive data, thereby mitigating the potential for data breaches and assuring adherence to privacy standards. Artificial intelligence (AI) deployed at the edge possesses the capability to do data processing and analysis without the need for data transmission via the internet.

3. *Enhanced Bandwidth Efficiency:* The utilization of edge AI facilitates the local processing of data, hence minimizing the volume of data that necessitates transmission to the cloud. This attribute is especially valuable in situations when bandwidth is constrained or incurs substantial expenses.

4. *Diminished Reliance on Network Connectivity:* Edge AI empowers devices to operate even in situations when internet connectivity is restricted or unavailable. This is crucial for applications in geographically isolated regions or in circumstances where dependable network connectivity cannot be assured.

5. *Improved User Experience:* In various applications such as gaming or augmented reality, the utilization of real-time processing at the edge can optimize the user experience by reducing latency in data processing, resulting in a more seamless and engaging interaction.

6. *Predictive Maintenance:* The implementation of AI models at the edge enables the real-time analysis of sensor data, facilitating the anticipation of equipment faults before to their occurrence. Preventing costly downtime is of utmost importance for industries such as manufacturing and utilities.

7. *Autonomous Vehicles:* Autonomous systems, such as drones, robotics, and self-driving automobiles, rely on edge AI as a crucial element.

These systems depend on the instantaneous analysis of sensor data to make rapid judgments.

8. *Healthcare and Telemedicine:* In the field of healthcare, the utilization of edge AI technology has proven to be advantageous in various aspects such as real-time patient monitoring, medical picture processing, and the provision of individualized treatment suggestions. This practice guarantees expedited response durations and maintains the confidentiality of patients.

9. *Smart Cities and IoT:* The integration of edge computing and artificial intelligence (AI) has the potential to facilitate the implementation of smart city programs. This technological combination enables the real-time analysis of data collected from diverse sensors and devices, thereby enhancing traffic management, energy efficiency, and public safety inside urban areas.

10. *Retail and Customer Engagement:* The application of edge AI in retail settings encompasses several functions such as inventory management, analysis of customer behavior, and provision of personalized shopping experiences. This has the potential to result in enhanced operational efficiency and heightened levels of customer satisfaction.

11. *Natural Language Processing (NLP):* Edge AI has the potential to be utilized in various applications such as speech recognition, language translation, and sentiment analysis. These tasks may be performed directly on devices such as smartphones or IoT devices, eliminating the need for dependence on cloud services. Consequently, this enables the implementation of natural language interfaces.

12. *Video Analytics:* The utilization of artificial intelligence (AI) in video analytics at the edge enables various applications such as surveillance, object recognition, and anomaly detection. This feature is especially advantageous in situations when the timely examination of video feeds is of utmost importance.

13. *Agriculture and Environmental Monitoring:* The utilization of edge AI in the domain of agriculture and environmental monitoring involves

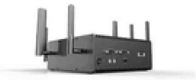
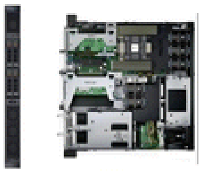
the processing of data obtained from sensors deployed in agricultural fields or environmental monitoring stations. This processing enables the extraction of valuable insights pertaining to crop health, soil conditions, and various environmental parameters.

14. *Energy Management:* Energy management can be enhanced through the utilization of edge AI, which enables the optimization of energy usage in various environments such as buildings and industrial settings. This is achieved by evaluating real-time data obtained from sensors and subsequently making necessary adjustments to heating, ventilation, and air conditioning systems.

15. *Intelligent Disaster Management:* The implementation of intelligent disaster management strategies plays a crucial role in mitigating the potential harm caused by impending calamities. Additionally, it is imperative to provide prompt and appropriate recommendations to the individuals affected in order to facilitate a swift recovery process. In contemporary times, the Internet of Things (IoT) has attained a high degree of sophistication and holds significant potential for offering considerable advantages in the context of crisis situations. The utilization of an Internet of Things (IoT) system that incorporates satellite communication and geographic information arrangements has been found to be effective in mitigating risks. The proposed approach entails proactive steps to avert incidents, timely dissemination of alerts, leveraging social media platforms for raising public consciousness, implementing relief and response strategies, as well as conducting search operations for missing individuals.

These applications exemplify the adaptability and potential influence of combining artificial intelligence with edge computing, establishing it as an essential technology for various industries and utilization scenarios. Table 3 deals with the consumer electronics devices for edge computing.

Table 3 Consumer electronics devices for edge computing

Product name	**Product picture**	**Features**	**Applications**	**Limitations**
Cisco Catalyst IR8100 Heavy Duty Series Routers	 *Source*: https://www.cisco.com/	Cisco Catalyst IR8100 Heavy Duty Series Routers are 5G, Rugged, and IP67-rated router. Can operate at −40° to 70 °C (−140°–158 °F) range. The IR8100 helps in control and real-time monitoring of industrial material to increase the operational efficiency	Cisco Catalyst IR8100 Heavy Duty Series Routers deliver private and public LTE and 5G	
Lenovo ThinkEdge SE50	 *Source*: https://news.lenovo.com/	ThinkEdge is built for various application that require higher data processing and analyzing at the edge end. It comes with Intel i5 or i7 processor and can operate on Windows 10 IoT LTSC Ubuntu Server 20.04, Core 20	Is used for complicated data analysis and huge data processing	Can be operated only at 0–50 degree temperature range
Dell PowerEdge XR11	 *Source*: https://www.dell.com/	Dell PowerEdge XR11. This edge computing server can work in dusty environment and extreme temperature condition	To create future-ready data center in IoT edge computing system	
Eaton 5P UPS	 *Source*: https://www.eaton.com/in/en-us/catalog/backup-power-ups-surge-it-power-distribution/eaton-5p-ups.html	Easy installation, users can pair a 5P UPS with an Eaton Gigabit Network card—the first in the industry to comply with the UL and IEC cybersecurity standards—and the Eaton Intelligent Power Management (IPM) platform for seamless management of the connected equipment	Is used as power backup in network closets, server rooms, distributed IT, edge computing	
HPE Edgeline EL8000		HPE Edgeline EL8000 can work in tough environments, is mountable in various places (shallow half-depth racks, cabinets, compact telecommunication enclosures, standing against a	Used to support AI and machine learning in edge computing system	

Product name	Product picture	Features	Applications	Limitations
	Source: https://buy.hpe.com/	wall without real access). Capable of running AI and machine learning		
Microsoft azure Pro R	 *Source*: https://learn.microsoft.com/en-us/azure/databox-online/azure-stack-edge-pro-r-overview	This device is capable of working in various difficult environment. Nvidia T4 GPU is used to enable AI interfacing, data analysis, data filtering, and data processing	Used as machine learner before sending data to server, to react and analyze events of IoT, to modify data by removing personal data	
Schneider Electric's APC Smart-UPS Ultra	 *Source*: https://www.apc.com/	This is very compact as it can be mounted rack, tower, wall and ceiling/desk under mount and 3 kw single unit UPS	Used in distributed edge computing to get uninterrupted power	

6 Open Research Challenges for Edge Computing

Challenge 1: How and where to store huge amount of transaction, computing data in edge computing?

As the IoT-supported industry continues to grow, transactions, and computing data are increasing rapidly, necessitating their storage for future use. Storing large amounts of data at Edge servers poses a significant challenge. However, a framework is required to detect important data that needs to be stored.

Challenge 2: How to avoid exchanging of computation task of different edge devices from edge server?

The edge devices are growing rapidly and they don't have enough computation resources to compute their own computation task. According to many of the authors, computation task has to offload to nearby edge server to optimally do computation task and propose task offloading scheme. However, most schemes compute these tasks one by one based on their priority. When a large number of tasks arrive at the edge server, tasks queue up, waiting for the completion of previous tasks.

There is currently no framework or model in place to ensure that the solutions of computation tasks are sent back to the originating Edge Device from which they were offloaded.

Challenge 3: How to implement server switching in multi-edge-server edge computing-based IoT network for locomotive end devices?

Let's consider a scenario of Multi-Edge-Server edge computing, where End Devices (EDs) have the freedom to move between different locations. In this scenario, there are edge servers with limited storage capacity and coverage range. The challenge arises when an End Device (ED) moves out of the coverage range of the previously connected Edge server. In such a scenario, the ED needs to establish a connection with another Edge server deployed adjacent to the previously connected one.

Challenge 4: How to manage real-world testing of resource allocation scheme?

The majority of the strategies that have been devised for distributing resources have been successful. The efficacy and success rate of resource allocation models can be seen from the outcomes of simulations that are carried out virtually. Simulations are carried out to test different scenarios. The experience will be very different in the real world compared to the simulated model. In order to model prototypes of these resource allocation schemes, it was necessary to research real-world situations.

Challenge 5: How is processing power managed in the context of Industry 5.0, which is driven by artificial intelligence?

Currently, there exists issues pertaining to computing power that are being faced by both technological and industrial companies. The proliferation of Internet of Things (IoT) devices has resulted in the generation of a substantial volume of data. This data is subsequently utilized for the purpose of training artificial intelligence (AI) models using techniques such as machine learning (ML) and deep learning (DL). The requirement for a steady power source presents notable challenges for both product manufacturing and startup ventures. Developers are often discouraged by the significant power demands associated with training algorithms in numerous instances. Although ML and DL components exhibit remarkable precision, they require a greater number of processing cores compared to conventional graphics processing units (GPUs). Scholars have conducted investigations into diverse techniques and methodologies in order to augment machine learning (ML) and deep learning (DL) models in a broad spectrum of applications. Furthermore, the integration of cloud computing with artificial intelligence (AI) and the utilization of parallel system processing with AI are being implemented in order to enhance the efficiency of power usage.

Challenge 6: Which inquiry pertains to the manner in which artificial intelligence (AI) effectively addresses the technical obstacles encountered within the context of Internet of Things (IoT) at the edge?

Artificial intelligence (AI) effectively addresses the technical obstacles encountered in the context of Internet of Things (IoT) at the edge. A significant number of contemporary research-oriented models for edge computing in the Internet of Things (IoT) domain remain current. Nevertheless, there exists practical obstacles that must be addressed, one of which is effectively managing a network that is characterized by heterogeneity. The implication of IoT is the presence of a significant multitude of connections that necessitate communication with diverse networks situated at the periphery. Although there are presently efficient systems accessible for managing these network arrays, they frequently exhibit disadvantages in terms of intricacy and energy use.

Service-oriented architecture in the context of the Internet of Things (IoT) is now encountering notable obstacles pertaining to performance and cost constraints. The proliferation of linked devices across many applications has led to an increase in the complexity of scalability, data pre-processing, service delivery, and networking. Furthermore, the transmission of sensed data across diverse network infrastructures gives rise to persistent delays and communication challenges. Therefore, there exists an urgent want for advanced automated protocols capable of effectively managing data acquired from various devices and efficiently processing it within edge-based Internet of Things (IoT) systems. Moreover, the issue of establishing efficient collaboration across interconnected devices belonging to diverse entities continues to be a subject of ongoing research. It is of utmost importance to adequately address the aspects of identification and optimization within this particular environment. While the influence of newly introduced items on IoT at the edge is minor, the implementation of integrated methods such as a unified data structure is still necessary. AI-based methodologies utilize significant features of IoT data, yet encounter obstacles pertaining to intricacy and energy usage.

Challenge 7: How can Industry 5.0 effectively manage diverse implementation strategies?

Given the recent advancements in a variety of network technologies, AI has the potential to revolutionize any industry and shift it toward digitalization. However, a key challenge with AI lies in generating innovative ideas for its practical application. A strategic approach is crucial for success in a competitive business environment, which involves identifying areas of progress, discerning valuable targets, and learning from past failures, among other factors. To tackle these challenges effectively, company managers, supervisors, and technical teams must possess a comprehensive understanding of AI skills, advancements, and limitations, while also staying updated on current issues faced by AI. By staying informed about the latest developments in AI-related information and adapting their work practices accordingly, organizations can easily identify areas where AI can bring significant improvements.

Challenge 8: How does the integration of AI technology in Industry 5.0 impact product management?

Radio frequency identification (RFID) tags are employed in a multitude of industries, encompassing supply-chain management, product tracking, and delivery logistics. The aforementioned tags are affixed to merchandise throughout the transportation phase, while readers are strategically positioned throughout the entirety of the route in order to monitor and maintain a record of their movement. The utilization of Internet of Things (IoT) devices provides more adaptability in the positioning of readers, thereby guaranteeing smooth interoperability among various RFID-based solutions employed by diverse stakeholders. IoT devices are of significant importance within the retail industry since they fulfill essential functions such as showcasing product availability and ensuring the maintenance of precise stock records. Furthermore, the use of sensors and biosensors in conjunction with RFID technology holds the promise of facilitating oversight of production processes and augmenting the overall quality of end products within the food business.

7 Conclusions

This chapter featured an investigation that began with understanding the vision pattern of AI-enabled edge computing skills and progressed to an analysis of its possible benefits in a variety of sectors. It also helps academics and experts who recognize the design construction and AI algorithms that use edge devices. This chapter examines a functional framework in depth, followed by a knowledge hierarchy for edge computing, object identification, intelligent sensors, learning, and analytics in intelligent IoT-edge-enabled systems. This chapter investigates the future era of AI-enabled IoT-edge paradigms that will be used to improve humanity's well-being. The wide citations of numerous publications and their practical applications, in particular, will be acknowledged as important resources for engineers and companies.

References

1. Xiong, X., Zheng, K., Lei, L., Lu, H.: Resource allocation based on deep reinforcement learning in IoT edge computing. IEEE J. Sel. Areas Commun., 1–1 (2020). https://doi.org/10.1109/JSAC.2020.2986615
2. Zhang, Y., Di, B., Zheng, Z., Lin, J., Song, L.: Joint data offloading and resource allocation for multi-cloud heterogeneous mobile edge computing using multi-agent reinforcement learning. In: 2019 IEEE Global Communications Conference (GLOBECOM), pp. 1–6 (2019). https://doi.org/10.1109/GLOBECOM38437.2019.9013596
[Crossref]
3. Khanfor, A., Hamadi, R., Ghazzai, H., Yang, Y., Haider, M.R., Massoud, Y.: Computational resource allocation for edge computing in social Internet-of-Things. In: 2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 233–236 (2020). https://doi.org/10.1109/MWSCAS48704.2020.9184663
[Crossref]
4. Liu, X., Yu, J., Wang, J., Gao, Y.: Resource allocation with edge computing in IoT networks via machine learning. IEEE Internet Things J. **7**(4), 3415–3426 (2020). https://doi.org/10.1109/JIOT.2020.2970110
[Crossref]
5. Tianqing, Z., Zhou, W., Ye, D., Cheng, Z., Li, J.: Resource allocation in IoT edge computing via concurrent federated reinforcement learning. IEEE Internet Things J. **9**(2), 1414–1426 (2022). https://doi.org/10.1109/JIOT.2021.3086910
[Crossref]

6. Guo, W., Chang, Z., Guo, X., Jayakody, D.N.K., Ristaniemi, T.: Resource allocation for edge computing-based blockchain: a game theoretic approach. In: 2020 IEEE International Conference on Communications Workshops (ICC Workshops), pp. 1–6 (2020). https://doi.org/10.1109/ICCWorkshops49005.2020.9145340
[Crossref]

7. Liu, X., Yu, J., Gao, Y.: Multi-agent reinforcement learning for resource allocation in IoT networks with edge computing. China Commun. **17**(9), 220–236 (2020)
[Crossref]

8. Fan, Y., Wang, L., Wu, W., Du, D.: Cloud/edge computing resource allocation and pricing for mobile blockchain: an iterative greedy and search approach. IEEE Trans. Comput. Soc. Syst. **8**(2), 451–463 (2021). https://doi.org/10.1109/TCSS.2021.3049152
[Crossref]

9. Li, X., Zhao, L., Yu, K., Aloqaily, M., Jararweh, Y.: A cooperative resource allocation model for IoT applications in mobile edge computing. Comput. Commun. **173** (2021). https://doi.org/10.1016/j.comcom.2021.04.005

10. He, Y., Wang, Y., Qiu, C., Lin, Q., Li, J., Ming, Z.: Blockchain-based edge computing resource allocation in IoT: a deep reinforcement learning approach. IEEE Internet Things J. **8**(4), 2226–2237 (2021). https://doi.org/10.1109/JIOT.2020.3035437
[Crossref]

11. Xu, J., Hu, Z., Zou, J.: Computing Offloading and Resource Allocation Algorithm Based on Game Theory for IoT Devices in Mobile Edge Computing (2020). https://doi.org/10.24507/ijicic.16.06.1895
[Crossref]

12. Deng, S., Xiang, Z., Zhao, P., Taheri, J., Gao, H., Yin, J., Zomaya, A.: Dynamical resource allocation in edge for trustable IoT systems: a reinforcement learning method. IEEE Trans. Industr. Inform., 1–1 (2020). https://doi.org/10.1109/TII.2020.2974875

13. Alqerm, I., Pan, J.: Enhanced online Q-learning scheme for resource allocation with maximum utility and fairness in edge-IoT networks. IEEE Trans. Netw. Sci. Eng., 1–1 (2020). https://doi.org/10.1109/TNSE.2020.3015689

14. Na, W., Jang, S., Lee, Y., Park, L., Dao, N.-N., Cho, S.: Frequency resource allocation and interference management in mobile edge computing for an Internet of Things system. IEEE Internet Things J., 1–1 (2018). https://doi.org/10.1109/JIOT.2018.2885348

15. Chen, M., Miao, Y., Gharavi, H., Hu, L., Humar, I.: Intelligent traffic adaptive resource allocation for edge computing-based 5G networks. IEEE Trans. Cogn. Commun. Netw., 1–1 (2019). https://doi.org/10.1109/TCCN.2019.2953061

16. An, X., Fan, R., Hu, H., Zhang, N., Atapattu, S., Tsiftsis, T.A.: Joint task offloading and resource allocation for IoT edge computing with sequential task dependency. ArXiv, abs/2110.12115. (2021)

17. Wang, C., Liang, C., Yu, F.R., Chen, Q., Tang, L.: Computation offloading and resource allocation in wireless cellular networks with mobile edge computing. IEEE Trans. Wirel.

Commun. **16**(8), 4924–4938 (2017). https://doi.org/10.1109/TWC.2017.2703901
[Crossref]

18. Qin, Z., Cheng, Z., Lin, C., Lu, Z., Wang, L.: Optimal workload allocation for edge computing network using application prediction. Wirel. Commun. Mob. Comput. **2021**, 1–13 (2021). https://doi.org/10.1155/2021/5520455
[Crossref]

19. Xu, J., Palanisamy, B., Ludwig, H., Wang, Q.: Zenith: utility-aware resource allocation for edge computing. In: 2017 IEEE International Conference on Edge Computing (EDGE), pp. 47–54 (2017). https://doi.org/10.1109/IEEE.EDGE.2017.15
[Crossref]

20. Zhao, L., Wang, J., Liu, J., Kato, N.: Optimal edge resource allocation in IoT-based smart cities. IEEE Netw. **33**(2), 30–35 (2019). https://doi.org/10.1109/MNET.2019.1800221
[Crossref]

21. Mahmood, A., Hong, Y., Ehsan, M.K., Mumtaz, S.: Optimal resource allocation and task segmentation in IoT enabled mobile edge cloud. IEEE Trans. Veh. Technol. **70**(12), 13294–13303 (2021). https://doi.org/10.1109/TVT.2021.3121146
[Crossref]

22. Ren, J., Yu, G., Cai, Y., He, Y.: Latency optimization for resource allocation in mobile-edge computation offloading. IEEE Trans. Wirel. Commun. **17**(8), 5506–5519 (2018). https://doi.org/10.1109/TWC.2018.2845360
[Crossref]

23. Liao, H., et al.: Learning-based context-aware resource allocation for edge-computing-empowered industrial IoT. IEEE Internet Things J. **7**(5), 4260–4277 (2020). https://doi.org/10.1109/JIOT.2019.2963371
[Crossref]

24. Tran, T.X., Pompili, D.: Joint task offloading and resource allocation for multi-server mobile-edge computing networks. IEEE Trans. Veh. Technol. **68**(1), 856–868 (2019). https://doi.org/10.1109/TVT.2018.2881191
[Crossref]

25. Wang, P., Yao, C., Zheng, Z., Sun, G., Song, L.: Joint task assignment, transmission, and computing resource allocation in multilayer mobile edge computing systems. IEEE Internet Things J. **6**(2), 2872–2884 (2019). https://doi.org/10.1109/JIOT.2018.2876198
[Crossref]

26. Nath, S., Li, Y., Wu, J., Fan, P.: Multi-user multi-channel computation offloading and resource allocation for mobile edge computing. In: ICC 2020–2020 IEEE International Conference on Communications (ICC), pp. 1–6 (2020). https://doi.org/10.1109/ICC40277.2020.9149124
[Crossref]

27. Satyanarayanan, M.: The emergence of edge computing. Computer. **50**(1), 30–39 (2017). https://doi.org/10.1109/MC.2017.9
[Crossref]

28. Garcia, C.G., Nunez-Valdez, E.R., Vicente Garcia-Diaz, B., PelayoGBustelo, C., Lovelle, J.M.C.: A review of artificial intelligence in the Internet of Things. Int. J. Interact. Multimed. Artif. Intell. **5**(4), 9–20 (2018)

29. Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., Ayyash, M.: Internet of Things: a survey on enabling technologies, protocols, and applications. IEEE Commun. Surv. Tutor. **17**(4), 2347–2376 (2015)
[Crossref]

30. Lin, J., Wei, Y., Zhang, N., Yang, X., Zhang, H., Zhao, W.: A survey on Internet of Things: architecture, enabling technologies, security and privacy, and applications. IEEE Internet Things J. **4**(5), 1125–1142 (2017)
[Crossref]

31. Samie, F., Bauer, L., Henkel, J.: From cloud down to things: an overview of machine learning in Internet of Things. IEEE Internet Things J. **6**(3), 4921–4934 (2019)
[Crossref]

32. Altexsoft: IoT Architecture: The Pathway from Physical Signals to Business Decisions (2020). https://www.altexsoft.com/blog/iot-architecture-layers-components/

33. Lin, Y.C.: Development of advanced manufacturing cloud of things (AMCoT)—a intelligence manufacturing platform. IEEE Robot. Autom. Lett. **2**(1), 1809–1816 (2017)
[Crossref]

34. Chen, C.C.: A novel automated construction scheme for efficiently developing cloud manufacturing services. IEEE Robot. Autom. Lett. **3**(3), 1378–1385 (2018). https://doi.org/10.1109/LRA.2018.2799420
[MathSciNet][Crossref]

35. Gupta, H., Dastjerdi, A.V., Ghosh, S.K., Buyya, R.: iFogSim: a toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. J. Softw. Pract. Exp. **47**(9), 1275–1296 (2015)
[Crossref]

36. Zhou, B., BuYya, R.: Augmentation techniques for mobile cloud computing: a taxonomy, survey, and future directions. ACM Comput. Surv. **51**(1) (2018). https://doi.org/10.1145/3152397

37. Ranjan, R., Rana, O., Nepal, S., Yousif, M., James, P., Wen, Z.: The next grand challenges: integrating the Internet of Things and data science. IEEE Cloud Comput. **5**(3), 12–26 (2018)
[Crossref]

38. Gubbi, J., BuYya, R., Marusic, S., Palaniswami, M.: Internet of Things (IoT): a vision, architectural elements, and future directions. Futur. Gener. Comput. Syst. **29**(7), 1645–1660 (2013)
[Crossref]

39. Ungureanu, A.V.: The transition from Industry 4.0 to Industry 5.0. The 4Cs of the global economic change. In: Nastase, C. (ed.) Lumen Proceedings: Vol. 13. 16th Economic International Conference NCOE 4.0 2020, pp. 70–81 (2020)
[Crossref]

40. Adi, E., Anwar, A., Baig, Z., Zeadally, S.: Machine learning and data analytics for the IoT. Neural Comput. & Applic. **32**, 16205–16233 (2020) [Crossref]

41. Sharma, I., Garg, I., Kiran, D.: Industry 5.0 and smart cities: a futuristic approach. Eur. J. Mol. Clin. Med. **07**(08), 2515–8260 (2020)

42. Aslam, F., Aimin, W., Li, M., Rehman, K.U.: Innovation in the era of IoT and Industry 5.0: absolute innovation management (AIM) framework. Information. **11**, 124 (2020). https://doi.org/10.3390/info11020124 [Crossref]

43. Nahavandi, S.: Industry 5.0—a human-centric solution. Sustainability. **11**, 4371 (2020). https://doi.org/10.3390/su11164371 [Crossref]

44. Qiu, T., Zhao, Z., Zhang, T., Chen, C., Philip Chen, C.L.: Underwater Internet of Things in smart ocean: system architecture and open issues. IEEE Trans. Industr. Inform. **16**, 1551–3203 (2019) (c) IEEE

45. Ozdemir, V., Hekim, N.: Birth of Industry 5.0: making sense of big data with artificial intelligence, 'The Internet of Things' and next-generation technology policy. OMICS J. Integr. Biol. **22**(1)., Mary Ann Liebert, Inc (2018). https://doi.org/10.1089/omi.2017.0194

46. Chander, B., Kumaravelan, G.: Cyber security with AI—part I. In: The "Essence" of Network Security: An End-to-End Panorama, pp. 147–171. Springer, Singapore (2021) [Crossref]

47. Skobelev, P.O., Borovik, S.Y.: On the way from Industry 4.0 to Industry 5.0: from digital manufacturing to digital society. Int. Sci. J. **2**(6), 307e311 (2017)

48. Chander, B., Pal, S., De, D., Buyya, R.: Artificial intelligence-based Internet of Things for Industry 5.0. In: Pal, S., De, D., Buyya, R. (eds.) Artificial Intelligence-Based Internet of Things Systems Internet of Things. Springer, Cham (2022). https://doi.org/10.1007/978-3-030-87059-1_1 [Crossref]

49. Bhattacharyya, S., Athithan, S., Pal, S., Sarkar, B., Akila, D., Chowdhury, S., Chandran, K., Gurusamy, S.: An IoT-enabled intelligent and secure manufacturing model using blockchain in hybrid cloud communication system. Secur. Commun. Netw. **2023**, Article ID 7556728, 12 pages (2023). https://doi.org/10.1155/2023/7556728

50. Pal, S., Jhanjhi, N.Z., Abdulbaqi, A.S., Akila, D., Alsubaei, F.S., Almazroi, A.A.: An intelligent task scheduling model for hybrid Internet of Things and cloud environment for big data applications. Sustainability. **15**, 5104 (2023). https://doi.org/10.3390/su15065104 [Crossref]

51. Tuan, N.A., Akila, D., Pal, S., Sarkar, B., Tran, T.K., Mothilal Nehru, G., Le, D.-N.: Dynamic data optimization in IoT-assisted sensor networks on cloud platform. Comput. Mater. Contin. **72**(1) (2022). https://doi.org/10.32604/cmc.2022.024096

52. Pal, S., VijayKumar, H., Akila, D., Jhanjhi, N.Z., Darwish, O.A., Amsaad, F.: Information-centric IoT-based smart farming with dynamic data optimization. Comput. Mater. Contin. **74**(2) (2023). https://doi.org/10.32604/cmc.2023.029038

53. Mukherjee, D., Ghosh, S., Souvik Pal, D., Akila, N.Z., Jhanjhi, M.M., AlZain, M.A.: Optimized energy efficient strategy for data reduction between edge devices in cloud-IoT. Comput. Mater. Contin. **72**(1) (2022). https://doi.org/10.32604/cmc.2022.023611

54. Pal, S., Jhanjhi, N.Z., Abdulbaqi, A.S., Akila, D., Almazroi, A.A., Alsubaei, F.S.: A hybrid edge-cloud system for networking service components optimization using the Internet of Things. Electronics. **12**(3), 649 (2023). https://doi.org/10.3390/electronics12030649
[Crossref]

OceanofPDF.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
S. Pal et al. (eds.), *IoT Edge Intelligence*, Internet of Things: Technology, Communications and Computing
https://doi.org/10.1007/978-3-031-58388-9_5

Service Provisioning at the Edge: An AI Approach Based on Policies

Zakaria Maamar[1] ✉, Amel Benna[2] ✉, Noura Faci[3] ✉ and Dalal Direm[4] ✉
(1) University of Doha for Science and Technology, Doha, Qatar
(2) Research Center for Scientific and Technical Information, Algiers, Algeria
(3) University of Lyon, UCBL, CNRS, INSA Lyon, LIRIS, Lyon, France
(4) High School of Computer Science, Algiers, Algeria

✉ **Zakaria Maamar (Corresponding author)**
Email: zakaria.maamar@udst.edu.qa

✉ **Amel Benna**
Email: abenna@cerist.dz

✉ **Noura Faci**
Email: noura.faci@univ-lyon1.fr

✉ **Dalal Direm**
Email: id_direm@esi.dz

Keywords Cloud – Edge – ODRL – Policy – Service

1 Introduction

Despite the multiple benefits of cloud computing like pay-per-use and dynamic scaling up/down, many ICT practitioners question its appropriateness for applications that expose strict latency and security non-functional requirements. Transferring data to (usually distant) clouds could take time because of high latency,[1] could be subject to interceptions, alterations, and misuses when transferred over public communication channels, and would depend on network availability and reliability. To satisfy these non-functional requirements, the trend nowadays is to couple cloud computing with edge[2] computing [5, 11, 19]. Simply put, edge computing is to provision computation, communication, and/or storage resources "next" (or nearby) to where data are collected reducing its transfer time and avoiding its long exposure to unnecessary risks. Contrasting cloud to edge, cloud means *more* resources, *more* reliability, and *more* latency, while edge means *less* resources, *less* reliability, and *less* latency [9]. According to Statista,[3] the global edge computing market size is estimated to $274bn in 2025. And, Computer Society considers synthetic intelligence at the edge (AI@side) among the top 10 technology trends in 2021.[4]

Whether cloud or edge, both embrace Anything-as-a-Service (XaaS) as an operation model allowing to relieve persons and organizations from the "burden" of owning and managing resources and shifting this "burden" to cloud and/or edge providers in return of a fee [14]. Software-aaS, Platform-aaS, and Infrastructure-aaS models are widely used to demonstrate XaaS. However, resource exposure through services does not offer the means nor guidelines for setting who is either permitted to use or prohibited from using some services with respect to a specific category of end-users, a specific period of time, a specific location, specific access rights, for example. This exposure even overlooks who must use services and what must be used. To address this lack of means and guidelines, we resort to the Open Digital Rights Language (ODRL, [21]) that would offer a fine-grained control over the usage of services. Briefly, ODRL expresses that *"something is permitted, forbidden, or obliged, possibly limited by some constraints"* [21]. ODRL is typically adopted in industries like eBook and online media where managing legal access to digital content is a concern.

In line with the current trend of reinforcing the role of edge computing in enabling the deployment of data-intensive applications (e.g., augmented reality and video analytics) [4, 10, 16], we gear the current work toward first shedding light on the roles that edge could play in this deployment and second defining necessary ODRL policies that would ensure the successful completion of these roles. For instance, a first role for edge is to back clouds when latency and security non-functional requirements are strict, and a second role for edge is to back peers when offloading becomes a must. What services would be anchored to these roles and what ODRL policies would control these services are examples of questions that we raise and address in this chapter. Our contributions are, but not limited to, (i) identification of roles that edge would take over, (ii) identification of services that edge exposes per role, (iii) refinement of services according to particular functional and non-functional requirements, (iv) access management to services using ODRL policies, and (v) technical demonstration of edges taking over roles and responding to services' demands in compliance with these ODRL policies. The rest of this chapter is organized as follows. Section 2 is an overview of cloud and edge computing and ODRL and also presents a case study. Section 3 discusses the different roles that edge computing could take over to provision services in compliance with the policies that constrain these roles. Technical details about implementing these roles are also presented in this section. Concluding remarks and future work are presented in Sect. 4.

2 Background

This section consists of 3 parts. The first 2 parts define cloud and edge computing as well as ODRL. The last part suggests a case study shedding light on the support that edge computing could offer to cloud computing.

2.1 Cloud and Edge Computing

Known for its capacity to provision services anytime, anywhere, cloud computing reduces the burden of in-house management of hardware and software resources on organizations. Indeed, the most popular cloud service models, SaaS, PaaS, and IaaS, allow organizations to focus on their core activities leaving technical concerns to cloud providers [14]. However, it was reported on many occasions the limited appropriateness of cloud computing for some real-time and critical applications where data freshness and security are major concerns [18, 23]. To address this limited appropriateness, an option is to process data at the edge, i.e., close to, of where it is collected and stored avoiding its transfer to distant clouds. This is known as edge computing.

Despite the separate niches of applications that each targets, De Donno et al. note the ongoing confusion between cloud and edge [5]. As stated in the introduction, both cloud's and edge's operation models provision services in return of a fee aiming to improve the quality-of-service and quality-of-experience of what is offered to users [20]. To mitigate this confusion, Yahya et al. propose 3 exclusive scenarios about the potential recipients of data during service provisioning [22]: cloud, only; edge, only; or both (and in what order: cloud first, then edge; edge first, then cloud; or concurrently). The selection of a scenario is dependent on 6 criteria referred to as *frequency*, *sensitivity*, *freshness*, *time*, *volume*, and *criticality*.

In [13], Masip-Bruin et al. introduce a layered fog-to-cloud (F2C) architecture, with clouds layer, intermediate cloud/fog systems layer, and edge fog devices layer. This architecture combines the strengths of both cloud and fog concepts while addressing their weaknesses. However, as the authors point out, this layered model demands a novel coordinated management strategy that drives new challenges like identifying cloud/fog capacity, coordinating the interactions among the different layers involved in the execution of the service, scheduling service execution to decide how a service's individual functions are split into the different fog layers and how each layer processes the services, and addressing security and reliability issues.

In [1], a solution called Policy Enforcement Fog Module (PEFM) performs local and remote policy enforcement for sensitive IoT data. PEFM is integrated into fog node's middleware to be the first processing point for raw sensing data that acts on data as close as and as soon as they are data generated. Data to protect are associated with a set of privacy policies that determine usage rules for these data where the Extensible Access Control Markup Language (XACML) is used to specify and evaluate access control policies.

While the literature on intertwining cloud and edge together is abundant [1, 22] and [23], there are not, to the best of our knowledge, studies that formally (i) define the roles of edge in making this intertwining succeed, (ii) expose services per role, and (iii) use policies to enforce this service provisioning.

2.2 Open Digital Rights Language

ODRL is an example of Rights Expression Languages (REL) that provides a flexible and interoperable information model, vocabulary, and encoding mechanisms to represent statements about the usage of assets. An asset is an identifiable resource of a collection of resources such as data/information, content/media, applications, and services. ODRL relies on policies to represent permitted and prohibited actions over a certain asset as well as required obligations that stakeholders must meet over this asset as well [6]. ODRL is adopted in different contexts and multiple fields and has recently been used to tackle the challenge of privacy of dynamic data like in IoT and mobile computing [3] and automatic extraction of rights even expressed in natural language [17].

Figure 1 is an excerpt of ODRL information model available at www.w3.org/TR/odrl-model whose constructs are as follows:

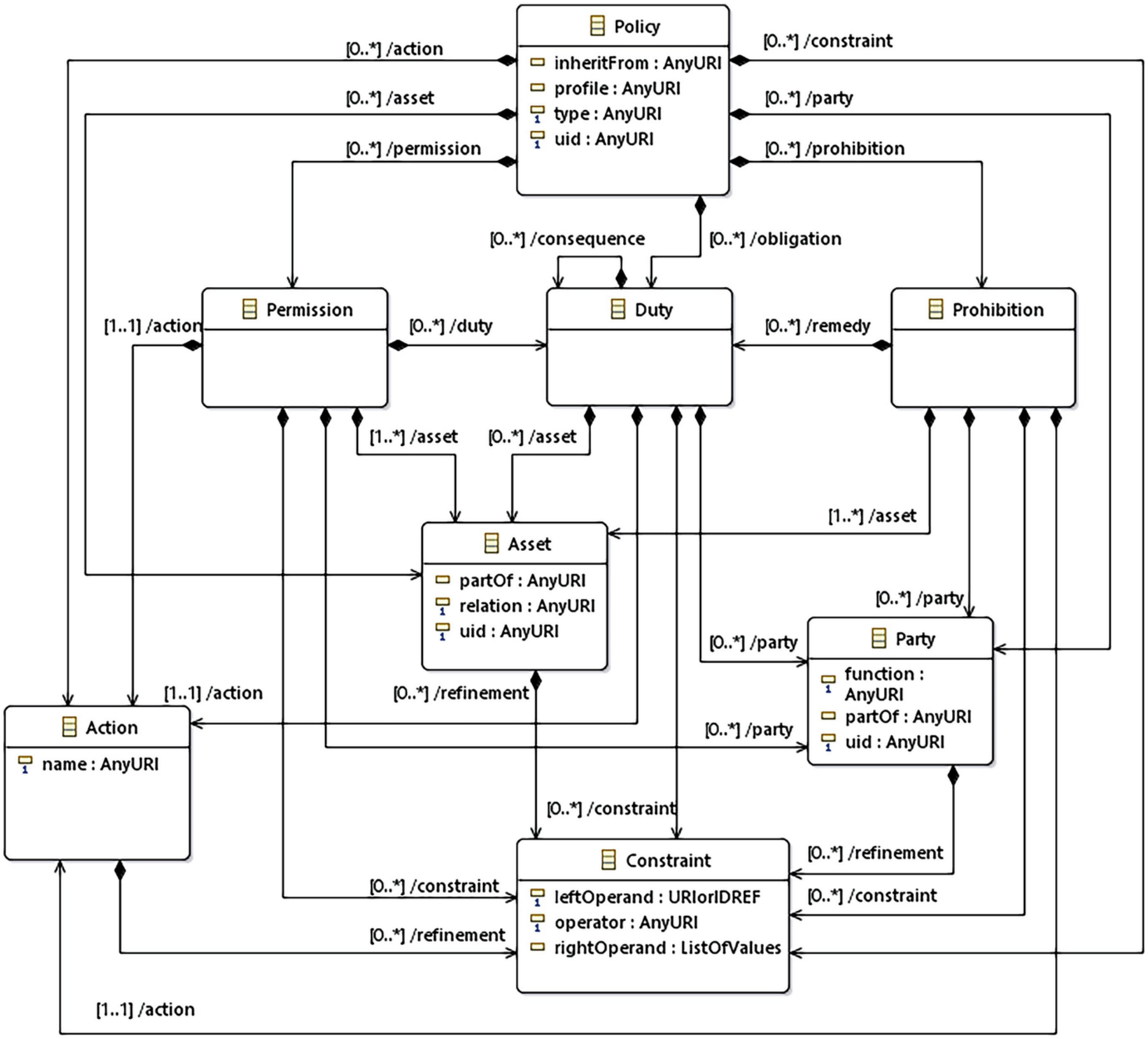

Policy
inheritFrom : AnyURI
profile : AnyURI
type : AnyURI
uid : AnyURI
[0..*] /action
[0..*] /asset
[0..*] /permission
[0..*] /constraint
[0..*] /party
[0..*] /prohibition
[0..*] /consequence
[0..*] /obligation
Permission
Duty
Prohibition
[1..1] /action
[0..*] /duty
[0..*] /remedy
[1..*] /asset
[0..*] /asset
[1..*] /asset
Asset
partOf : AnyURI
relation : AnyURI
uid : AnyURI
[0..*] /party
[0..*] /party
Party
function : AnyURI
partOf : AnyURI
uid : AnyURI
[1..1] /action
[0..*] /refinement
[0..*] /party
Action
name : AnyURI
[0..*] /constraint
[0..*] /refinement
[0..*] /constraint
Constraint
leftOperand : URIorIDREF
operator : AnyURI
rightOperand : ListOfValues
[0..*] /constraint
[0..*] /refinement
[1..1] /action

Fig. 1 Excerpt of ODRL information model diagram

- Policy could include one to many permission, prohibition, or duty rules. First, permission allows an action over an asset if all constraints are satisfied and if all duties are fulfilled. Second, prohibition disallows an action over an asset if all constraints are satisfied. Finally, duty forcibly exercises an action over an asset or not. It is fulfilled when all constraints are satisfied and have been exercised.
- Party is an entity or a collection of entities that could correspond to a person, a group of persons, organization, or agent. A party can fulfill different roles including assigner (issuer of the rule), assignee (recipient of the rule), informed party (to inform about all the uses), consented party (who obtains the consent), consenting party (issuer of the consent), compensated party (recipient of the compensation), and tracked party (indicating what is being tracked).
- Constraint is used either to refine components like action, party, or a collection of assets or to declare conditions applicable to a rule. A constraint could also be compared to one or many constraints using a logical operator like xone and all.

In Listing ?? serialized as JSON-LD, the assigner (line 6) in charge of an asset, *movie1* (line 5), refers to a policy labelled as agreement (line 3) consisting of 2 rules. The permission rule (line 4) is concerned with the action *display* (line 8) that an assignee, *smart TV* (line 7), will execute subject to satisfying the constraint that is not enabling the permission rule for more than 4 hours (lines 10–13). And, the prohibition rule (line 14) that is concerned with the action *digitize* (line 21) that is associated with the same assignee (line 20). Should the assignee execute this action, then the remedy (line 22) would consist of applying the action *anonymize* (line 22) to the asset that is *movie1* (line 23).

```
{"@context": "http://www.w3.org/ns/odrl.jsonld",
"uid": "http://example.com/policy:01",
"@type": "Agreement",
"permission": [{
    "target": "http://example.com/asset:movie1/",
    "assigner": "http://example.com/MovieParty:org:abc",
    "assignee": "http://example.com/smart-TV/",
    "action": "display",
    "constraint": [{
        "leftOperand": "meteredTime",
        "operator": "lteq",
        "rightOperand": {"@value": "P4H", "@type": "xsd:duration"},
        "unit": "https://www.wikidata.org/wiki/Q25235"}]}],
"prohibition": [{
    "target": "http://example.com/asset:movie1/",
    "assigner": "http://example.com/Movie1Party:org:abc",
    "assignee":"http://example.com/smart-TV/",
    "action": "digitize",
    "remedy": [{
        "action": "anonymize",
        "target": "http://example.com/asset:movie1/"}]}]
}
```

2.3 Case Study

With the latest ICT advances, many personal details about people are collected from multiple sources and, then, shared without strict control over those who receive and process these details for reasons like understanding people’s habits [24]. Our case study assumes a specialized company that runs many care centers for elderly people throughout the city. To ensure proper follow-up of the centers’ patients, many details are collected at the level of centers and then processed and/or stored at the same level and/or the headquarter level as well. Because of the sensitivity and criticality of some details, the centers are equipped with edge facilities, while the headquarter is equipped with cloud facilities. All the facilities exchange data according to specific pre-defined flows whether edge-to-edge, edge-to-cloud, cloud-to-cloud, or cloud-to-edge.

According to [8], the 5 most important smart tech-devices for senior safety include smart home hub, smart home sensors, smart lights, smart medication dispensers, and smart stove shutoff. Let us consider the situation where a center’s patients get together every afternoon to watch movies in the living room. Each patient is expected to have her own dispenser when she is not in her room extracting details like medication frequency and doses from the edge facilities and updating the same facilities so that the medical staff is aware of what has been administered, at what time, etc. Prior to the viewing session, the smart TV accesses the same edge facilities to synchronize with all the attendees’ dispensers so that reminders are displayed on the screen and, if required, alerts are sent to the medical staff. While this synchronization is mandatory, patients’ and medicines’ details could be used by third parties to develop targeted healthcare awareness campaigns for these patients that could make them subscribe to programs announced in these campaigns. This shows the importance of safeguarding these details prior to their transfer, whether as is or aggregated to edge and/or cloud facilities.

While the paragraph above assumes a well-defined data flow from patients’ wearable devices to edge facilities and/or cloud facilities and between these facilities, there are other scenarios where data latency and confidentiality are a serious concern. An example is a hospital’s around-the-clock emergency department, which deals with genuine life-threatening cases (e.g., breathing difficulties, severe allergic reactions, and consequent high blood pressure), where patients may have only moments before a dip in vital signs that could end in a catastrophic crash. In such cases, readings from patients’ wearable devices need to make it to the doctors within split-seconds time frame; otherwise, life could easily be lost. Ensuring the freshness of these readings would call for edge facilities supporting real-time analytics and quicker access to patients’ details.

3 Roles of Edges in Service Provisioning

This section identifies the roles that edge nodes could take over depending on future applications’ functional and non-functional requirements. Along with this identification, the section defines service provisioning across the different roles and then, per role. In the following, edge facility and edge node are interchangeably used.

3.1 Edges' Prescribed Roles

3 roles can edge node take over: solo, cloud support, and edge support. Duties of each role are discussed below:

Solo role. It would assume that future applications are satisfied with the services that edge nodes provision despite the fact that these nodes are known for offering less resources and being less reliable than cloud nodes (Fig. 2). With respect to the case study, the smart TV acting as an edge node could host a monitoring application that collects and analyzes attendees' heartbeats and notifies the medical staff if required. Edge nodes are self-sufficient and do not require any interactions with third parties whether peers or not.

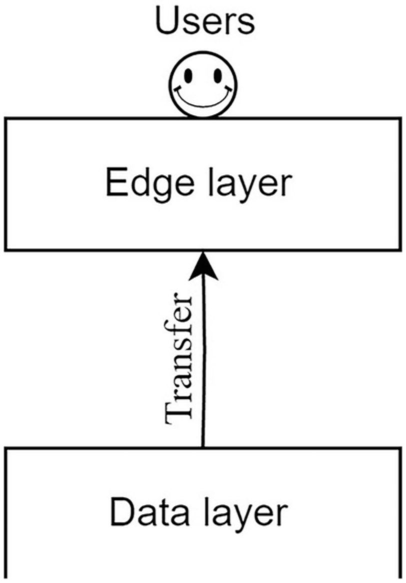


Fig. 2 Edge node taking over the solo role

Cloud-support role. It would assume that future applications have strict latency and security non-functional requirements making cloud nodes inappropriate and hence, requiring the participation of edge nodes in satisfying these requirements by provisioning necessary services at the "edge" of the data layer (Fig. 3a). With respect to the case study, an example of application would be smart wrist devices that read patients' vitals and immediately share the readings through edge nodes with medical staff for action. Edge nodes' respective details are fed into cloud nodes for aggregation purposes. Whether at the edge or cloud layers, specific categories of users would interact with these layers according to their needs and requirements.

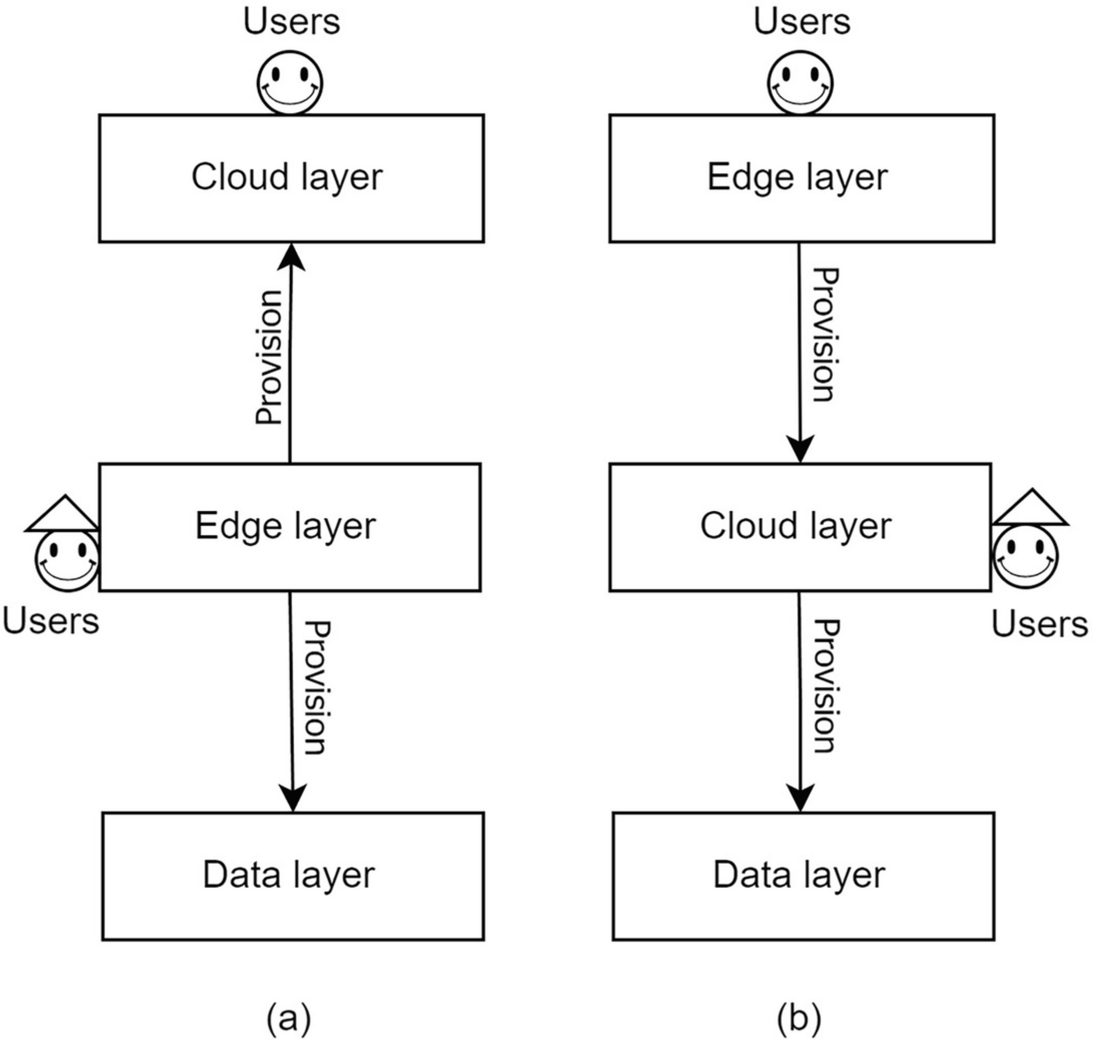

Users
Cloud layer
Provision
Users
Edge layer
Provision
Data layer
(a)
Users
Edge layer
Provision
Cloud layer
Users
Provision
Data layer
(b)

Fig. 3 Edge node taking over the cloud-support role

Contrarily to Fig. 3a where the edge layer sits between the data and cloud layers, Fig. 3b proceeds differently where the edge layer comes on top of the cloud layer, in fact at the end of the data-transfer sequence. This would be possible when future applications would relax the strict latency and security non-functional requirements making cloud nodes appropriate. Despite this relaxation, cloud nodes could always transfer raw/processed data to edge nodes depending on future applications' needs and requirements. By analogy with Fig. 3a, users would interact with the edge and cloud layers. With respect to the case study, an example of application would be smart wrist devices that back up their daily readings directly into cloud nodes that themselves channel these readings after processing to appropriate location-based edge nodes.

Peer-support role. It would assume that future applications allow edge nodes to engage in collaboration with peers when implementing what is usually known in the literature as offloading [7]. As per Fig. 4, when an edge node detects a potential increase in its load that could lead to performance degradation, the edge node could reach out to peers as per specific agreements that could be based on trust, for example [12]. With respect to the case study, an example of application would be the same ambient sensors that due to an increase in the frequency of monitoring would transfer more data to their assigned edge nodes that themselves transfer these data to peers due to limited storage capacity.

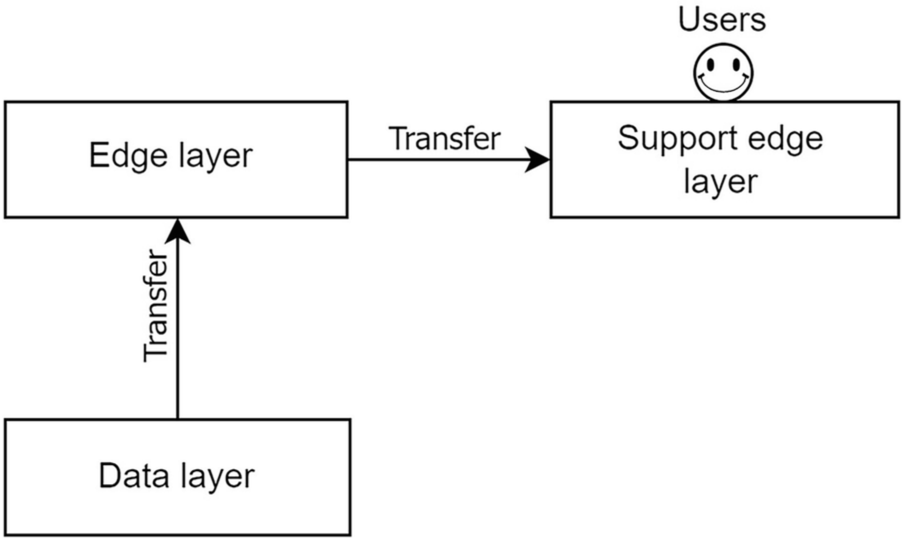


Fig. 4 Edge node taking over the peer-support role

3.2 Services Across all Roles

Independently of the roles that edge nodes would take over, i.e., solo, cloud support, or edge support, we consider specific services that would drive the interactions between an edge node and a data layer, cloud layer, or edge layer. We refer to these services as binding, interaction, and unbinding and entrust their control to a set of ODRL policies:

Binding service. To enable the interactions between an edge node and third parties namely, data and cloud layers and other edge nodes, 2 steps referred to as *authentication* and *conformance* are deemed necessary. As a result, we associate each step with an aspect that would fall into the binding service. When it comes to the *authentication* aspect, an edge node would require that third parties identify themselves in compliance with specific security guidelines that the edge node would set. We regulate the definition and use of these guidelines with 2 policies. In Listing ??, the first policy of type request (line 3) uses the action annotate (line 7) to first enrich the edge node with particular security details like protocol's type and/or minimum credential details and second "justify" the rationale (lines 10–13) of this enrichment that is making future third parties aware of what is needed to successfully interact with the edge node. The annotations that result out of the first policy are treated as assets (line 14) and will be part of the definition of the second policy reported in Listing ??. In this policy, there is a permission rule (line 4) that refers among many other things to the action use (line 8) for binding the data layer to the edge node and the obligation (line 9) on the data layer to use (line 10) the annotations linked to the edge node (line 11).

```
{"@context": "http://www.w3.org/ns/odrl.jsonld",
"uid": "http://example.com/EdgeNode/policy:111",
"@type": "Request",
"permission": [{
    "target": "http://example.com/Edge/EdgeNode:003",
    "assignee": "http://example.com/EdgeNodeOwner:org",
    "action": [{
        "rdf:value": {"@id": "odrl:annotate"},
        "refinement": [{
        "leftOperand": "purpose",
        "operator": "eq",
        "rightOperand": {"Data layer use", "@type": "xsd:string"},
        "comment": "Data layer use"}]}],
    "outputs": "http://example.com/EdgeNode/annotations"
}]}
```

```
{"@context": "http://www.w3.org/ns/odrl.jsonld",
"uid": "http://example.com/EdgeNode/policy:002",
"@type": "Agreement",
"permission": [{
    "target": "http://example.com/Edge/EdgeNode:003",
    "assigner": "http://example.com/EdgeNodeOwner:org",
    "assignee": "http://example.com/DataLayerOwner",
    "action": "use",
    "duty": [{
        "action":"use",
      "target": "http://example.com/EdgeNode/annotations"}]
}]}
```

After successful authentication where the edge node would confirm the communication channels to use with data/cloud/edge layers so that it receives their instructions and/or data, the *conformance* aspect becomes active. Here, the edge node is expected to comply with the purpose that their owner sets and announces to all layers. This could be enforced through a prohibition rule preventing the edge node from performing any self- or third-party-initiated action that could result into provisioning other services than what is mentioned in the purpose. For illustration, Listing ?? includes details like the offer policy (line 3) defined by the edge node's owner (line 6) and targeting a wider audience of potential third parties (i.e., no specific assignee is set) including the edge node itself, and the prohibition rule (line 4) that prevents using (line 8) the node edge (line 5) for a different purpose than storage (lines 10–13).

```
1  {"@context": "http://www.w3.org/ns/odrl.jsonld",
2  "uid": "http://example.com/Edge/policy:003",
3  "@type": "Offer",
4  "prohibition": [{
5      "target": {"http://example.com/Edge/EdgeNode:003"
           }
6      "assigner": "http://example.com/EdgeNodeOwner:org
           ",
7      "action": [{
8          "rdf:value": {"@id": "odrl:use"},
9          "refinement": [{
10             "leftOperand": "purpose",
11             "operator": "neq",
12             "rightOperand": "https://www.wikidata.org
                   /wiki/Q66221",
13             "comment": "Storage, only"}]}]
14 }]}
```

Interaction service. After successful invocation of the binding service, an edge node would begin interacting with data/cloud/edge layers as per the announced purpose. Two critical aspects need to be maintained during interactions namely, *performance* and *integrity*. First, the performance aspect identifies potential parameters like latency and availability that could impact the quality of interactions. For illustration, Listing ?? includes details like the prohibition rule (line 4) that sets the type (lines 11–14) and volume (lines 16–19) of data to transfer from the assignee that is the cloud layer (line 7) to the edge node.

```
1  {"@context": "http://www.w3.org/ns/odrl.jsonld",
2  "uid": "http://example.com/policy:004",
3  "@type": "Agreement",
4  "prohibition": [{
5      "target": "http://example.com/EdgeNode:001",
6      "assigner": "http://example.com/EdgeNodeOwner:org",
7      "assignee": "http://example.com/CloudLayerOwner:001",
8      "action": [{
9          "rdf:value": {"@id": "odrl:use"},
10         "refinement": [{
11             "leftOperand": "file format",
12
13
14             "operator": "eq",
15             "rightOperand": {"https://www.wikidata.org/wiki/Q86920"},
16             "comment": "file containing plain text"
17             },{
18             "leftOperand": "percentage",
19             "operator": "gt",
20             "rightOperand":{"@value": "70.00","@type": "xsd:decimal"},
21             "comment": "70% of exercising the action"}]}]
22 }]}
```

Second, the integrity aspect manages data in terms of who can update what, when, where, how, and why. For illustration, Listing ?? includes details like the target that is the data stored by the edge node (line 4), the assigner that is the edge node’s owner (line 5), and 2 rules. The prohibition rule (line 6) prevents specific actions like modify and attribute (line 8) from being carried out where the first action concerns the edge node and the second action concerns those that have been designated by the edge node (line 9). And, the permission rule (line 10) authorizes the data’s owner that is the data layer to carry out the action modify (line 12).

```
1  {"@context": "http://www.w3.org/ns/odrl.jsonld",
2  "uid": "http://example.com/policy:005",
3  "@type": "Request",
4  "target": "http://example.com/EdgeNodeData:003",
5  "assigner": "http://example.com/EdgeNodeOwner:org",
6  "prohibition": [{
7      "assignee": "http://example.com/EdgeNodeOwner:org",
8      "action": ["modify", "attribute"],
9      "attributedParty":"http://example.com/CloudNode:004"}],
10 "permission": [{
11     "assignee": "http://example.com/DatalayerOwner",
12     "action": "modify"}]}
```

Unbinding service. Upon the decision of closing the communication channels established as a result of the binding, unbinding the edge node from data/cloud/edge layers in a systematic way is taken care by an aspect that we refer to as *removal*. This aspect aims at avoiding “orphan data” situations by specifying, for instance, the data that need to be deleted first, the acceptable delays of deleting the data, and any other detail judged important for the layer requesting the unbinding. For illustration, Listing ?? includes details like the obligation rule (line 4) that imposes on the edge node (line 5) to delete (line 8) all the data (line 5) it received from the data layer upon receipt of an unbinding request.

```
{"@context": "http://www.w3.org/ns/odrl.jsonld",
"uid": "http://example.com/EdgeNode/policy:006",
"@type": "Request",
"obligation": [{
    "target": "http://example.com/EdgeNodeData:003",
    "assignee": "http://example.com/EdgeNodeOwner:001",
    "action": [{
       "rdf:value": {"@id": "odrl:delete"},
       "refinement": [{
          "leftOperand": "purpose",
          "operator": "eq",
          "rightOperand":{"datalayer unbinds", "@type": "xsd:string"},
          "comment":"Datalayer unbinds from the edge node"}]}]
}]}
```

3.3 Service Adjustment Per Role

While in Sect. 3.2 we defined common services across the 3 roles, we proceed this time with refining these services using specialized aspects that would cater for each role that an edge node would take over:

Solo role. In conjunction with *performance* and *integrity* aspects that fall into the interaction service, we consider an additional aspect that is *collection*. During interaction, an edge node would be tasked of receiving data from data sources comprising the data layer for multiple purposes like aggregation and even storage. These purposes are set according to future applications' needs and requirements. In line with the previous policies, we regulate the *collection* aspect using, as per Listing ??, a permission rule (line 4) that allows to collect data, represented by the action include (line 11), from data sources treated as a first asset (lines 6–7), and store these data at the level of the edge node treated as a second asset (line 21). This collection happens every 2 days (lines 17–20) and for aggregation purposes (lines 13–16). It is worth noting that the existence of 2 assets in this policy is in line with how ODRL defines the action include. Finally, although not specified in the policy, it could happen that the data collected from the data sources would be deleted using the ODRL action move.

```
{"@context": "http://www.w3.org/ns/odrl.jsonld",
"uid": "http://example.com/EdgeNode/policy:007",
"@type": "Request",
"permission": [{
    "target": {
        "@type": "AssetCollection",
        "uid":http://example.com/DataLayersources:001"},
    "assigner": "http://example.com/DataLayerOwner:org",
    "assignee": "http://example.com/EdgeNodeOwner::001",
    "action": [{
        "rdf:value": {"@id": "odrl:include"},
        "refinement": [{
           "leftOperand": "purpose",
           "operator": "eq",
           "rightOperand":"https://www.wikidata.org/wiki/Q23007394",
           "comment": "aggregation"},
           {"leftOperand": "timeInterval",
           "operator": "eq",
           "rightOperand": {"@value": "P2D", "@type": "xsd:string"},
           "comment": "every 2 days"}]}],
    "output":"http://example.com/EdgeNode/collectedData:003"}]}
{"@type": "dc:Document",
"@id": "http://example.com/asset/datasource:004",
"dc:title": "Datasources",
"odrl:partOf": "http://example.com/DataLayersources:001"}
```

Cloud-support role. As per Sect. 3.1, edge nodes are placed either between the data and cloud layers (Fig. 3a) or at the end of the data-transfer sequence (Fig. 3b):

> To materialize Fig. 3a, 3 additional aspects, *collection*, *extraction*, and *notification*, are considered. We begin the discussion with the first two aspects that fall into the interaction service and are geared toward the data layer and cloud layer, respectively. Then, we complete the discussion with the last aspect that falls into the unbinding service and is geared toward the cloud layer, only.
>
> Although the *collection* aspect in both solo and cloud-support roles achieves the same objective, an edge node in the second role acts as a transit facility for the data submitted by the data sources comprising the data layer. This transit nature calls for specific ODRL policies allowing for instance, to filter the data to retain at the edge node level from those to transfer to the cloud layer with or without processing and/or delay. Filtering would consider the nature of data like sensitivity, urgency, and confidentiality. Taking advantage of the objective similarity of the *collection* aspect, we avoid defining a new policy for this aspect from scratch by inheriting from the already-defined collection (parent) policy in Listing ??. The obtained policy defined in Listing ?? uses first inheritFrom property (line 3) to extend rules from the parent policy and second a permission rule (line 7) that annotates (line 16) the collected data (lines 9–10) with focus here on the non-sensitive one (lines 12–15) for use by the cloud layer (lines 18–20). The annotations returned (line 22) aim at enriching data with any details that the cloud layer would deem necessary like data origin and whether or not the data have been processed by the edge node.

```
{"@context": "http://www.w3.org/ns/odrl.jsonld",
"uid": "http://example.com/policy:008",
"inheritFrom":"http://example.com/EdgeNode/policy:007
   ",
"@type": "Request",
"assigner": "http://example.com/EdgeDataOwner:org",
"assignee": "http://example.com/EdgeNodeOwner:org",
"permission": [{


   "target": {
      "@type": "AssetCollection",
      "source": "http://example.com/EdgeNode/
         collectedData:003",
      "refinement": [{
        "leftOperand": "product",
        "operator": "neq",
        "rightOperand":"https://www.wikidata.org/wiki
           /Q2587068",
        "comment": "non-sensitive data"}]},
    "action":[{"rdf:value": {"@id": "odrl:annotate" }
       ,
      "refinement": [{
        "leftOperand": "purpose",
        "operator": "eq",
        "rightOperand": {"@value":"cloud layer use",
           "@type": "xsd:string"},
        "comment": "cloud layer use"}]}],
    "output":"http://example.com/EdgeNode/
       cloudAnnotation:4"
}]}
```

Following the collection of data that could have been filtered and/or annotated by the edge node, the *extraction* aspect becomes active where data to submit to the cloud layer are identified in response to future applications' business requirements. The submission could be set according to frequency (regular *versus* sporadic), priority (urgent *versus* not-urgent), or any other criteria that would have been mentioned during data annotation by the edge node. For illustration, Listing ?? includes details like the permission (line 6) for the cloud layer (line 4) to collect, represented by the action move (lines 10), the annotated data (lines 7–9) from the edge node (line 3) during a specific time period (lines 17–25) that is between 6:00am and 9:00am UTC time. Part of this permission rule, there is an obligation rule (line 26) on the cloud layer to refer, represented with the action use (line 27), to the data annotations (line 28) set by the edge node as per Listing ?? (line 25).

```
{"@context": "http://www.w3.org/ns/odrl.jsonld",
"uid": "http://example.com/EdgeNode/policy:009",
"assigner": "http://example.com/EdgeNodeOwner:org",
"assignee": "http://example.com/CloudLayer:org",
"@type": "Agreement",
"permission": [{
   "target": {
   "@type": "AssetCollection",
   "uid": "http://example.com/EdgeNode/collectedData:
      003"}
   "action": [{"rdf:value": {"@id": "odrl:move"},
      "refinement": [{
        "leftOperand": "recipient",
        "operator": "gt",
        "rightOperand": {"http://example.com/
           cloudlayer:org"},
        "comment": "cloud layer location"
        },{
        "leftOperand": "dateTime",
        "operator": "gt",
        "rightOperand": {"@value":"T06:00Z", "@type":
            "xsd:dateTime"},
        "comment": "after 6:00AM"
        },{
        "leftOperand": "dateTime",
        "operator": "lteq",
        "rightOperand": {"@value":"T09:00Z", "@type":
            "xsd:dateTime"},
        "comment": "before 9:00AM"}]}],
    "duty": [{
       "action": "use",
       "target":"http://example.com/EdgeNode/
          CloudAnnotation:4"}]
}]}
```

Along with the *collection* and *extraction* aspects, we mentioned above the *notification* aspect that would be a direct consequence of activating the *removal* aspect as part of the unbinding. The edge node informs the cloud layer that the data layer has stopped submitting data following the decision of unbinding, which means additional (fresh) data transmission will not happen. The cloud layer could either continue interacting with the edge node making this node the sole data source or unbind from the edge node like the data layer did. For illustration, Listing ?? includes details like the obligation rule (line 4) on the edge node's owner (line 10) to inform (line 8) the cloud layer (line 9) when the data layer unbinds from this edge node (lines 12–15). Should the cloud layer maintain the binding with the edge node, then it would expect limited data transfer.

```
1  {"@context": "http://www.w3.org/ns/odrl.jsonld",
2  "uid": "http://example.com/EdgeNode/policy:010",
3  "@type": "Request",
4  "obligation": [{
5      "assigner": "http://example.com/ClodLayerOwner:001",
6      "target": "http://example.com/Cloudlayer:003",
7      "assignee": "http://example.com/EdgeNode:001",
8      "action": "inform",
9      "informedParty":"http://example.com/cloudLayer",
10     "informingParty":"http://example.com/EdgeNodeOwner:001",
11     "constraint"[{
12         "leftOperand": "event",
13         "operator": "eq",
14         "rightOperand": {"@value":"data layer unbinds", "@type": "xsd:string"},
15         "comment": "Data layer unbinds from the edge node"}]
16 }]}
```

Contrarily to Fig. 3a where the edge layer addresses concerns like latency and security, this layer in Fig. 3b is more concerned with satisfying specific functional requirements like making data available close to where users would be located. For instance, an edge node could refine the elderly persons' aggregated medical recommendations it had received from the cloud layer according to the latest measurements of these persons' vitals. To materialize Fig. 3b, we consider an additional aspect, *collection*, that would fall into the interaction service and be geared toward the cloud layer. Once again, the *collection* aspect would be treated differently from those described earlier. The edge node is at the disposal of the cloud layer. For illustration, Listing ?? includes the target that is the cloud layer’s data (lines 6–7), that is labelled as sensitive (lines 9–12) due to its medical nature in the healthcare field (lines 13–16) and the permission rule (line 4) to the edge node (line 18) to collect and store this data at its level (lines 22–25), represented by the action move (line 20), in response to the request of the cloud layer (lines 27–30).

```
{"@context": "http://www.w3.org/ns/odrl.jsonld",
"uid": "http://example.com/EdgeNode/policy:011",
"@type": "Request",
"permission": [{
   "target": {
    "@type": "AssetCollection",
    "source":http://example.com/CloudLayerData",
      "refinement": [{
        "leftOperand": "product",
        "operator": "eq",
        "rightOperand":"https://www.wikidata.org/wiki
           /Q2587068",
        "comment": "sensitive data"},
         {"leftOperand": "industry",
        "operator": "eq",
        "rightOperand":"https://www.wikidata.org/wiki
           /Q31207",
        "comment": "healthcare"}]},
    "assigner": "http://example.com/cloudLayerOwner:
       org",
    "assignee": "http://example.com/EdgeNodeOwner::00
       1",
    "action": [{
        "rdf:value": {"@id": "odrl:move"},
        "refinement": [{
            "leftOperand": "recipient",
            "operator": "eq",
            "rightOperand":"http://example.com/
               EdgeNode",
            "comment": "EdgeNode X"}]}],
    "constraint": [{
        "leftOperand": "event",
        "operator": "eq",
        "rightOperand":{"@value": "upon the request
           of the cloud","@type": "xsd:string"},
        "comment": "upon the request of the cloud"}]
}]}
```

Peer-support role. Exposing some similarities with Fig. 3b where the cloud layer "pushes" in a controlled way some data to the edge layer, an edge node taking over the peer-support role would accept (raw or processed) data from peers as part of what the community refers to as offloading. We shape data “push” with another *collection* aspect that would fall into the interaction service. This aspect becomes active when computing, storage, and/or communication demands exceed an edge node’s capacity, which not only could degrade its performance but undermine its capability of provisioning services, as well. To address such a situation, other peers could handle some of the edge node’s data load ensuring the satisfaction of applications’ non-functional requirements. With respect to the case study, it happens that wearable sensors increase the frequency of measuring a patient’s vitals with the risk of overwhelming the wrist watch with many details like blood pressure and heartbeat. Should this overwhelming arise, the watch would offload some of these details to the patient’s smartphone. For illustration, Listing ?? includes details like the permission (line 4) that is given to an edge node (line 6) to transfer some data (lines 8–9), represented by the action move (line 10), to the support edge layer (lines 12–15). This transfer is conditioned by an obligation (line 16) on the edge node to verify that its storage capacity has exceeded 85% (lines 21–24).

```
{"@context": "http://www.w3.org/ns/odrl.jsonld",
"uid": "http://example.com/EdgeNode/policy:012",
"@type": "Request",
"permission": [{
   "assigner": "http://example.com/EdgeNodeOwner:org",
   "assignee": http://example.com/EdgeNodeOwner:org",
   "target": {
      "@type": "AssetCollection",
      "uid": "http://example.com/EdgeNodeData:002"},
    "action": [{"rdf:value": {"@id": odrl:move"},
      "refinement": [{
        "leftOperand": "recipient",
        "operator": "eq",
        "rightOperand": "http://example.com/support-edge:001",
        "comment": "support edge layer"}]}],
    "duty": [{
        "target":"http://example.com/EdgeNodeStorageCapacity",
        "action": [{
           "rdf:value": {"@id": "odrl:use"},
           "refinement": [{
             "leftOperand": "percentage",
             "operator": "gt",
             "rightOperand":{"@value":"85.00","@type": "xsd:decimal"},
             "comment": "85 percent"}]}]}]
}]}
```

3.4 Discussions

Table 1 summarizes the way we coupled services/aspects to roles where ● indicates that a service/aspect applies to all roles without adjustment, ◇ indicates that a service/aspect applies to all roles with adjustment, and ○ indicates that a service/aspect applies to specific roles, only. Using this table, we can first highlight how to load an edge node with necessary services/aspects, so that it fulfills its prescribed role, and second identify the variations and uniqueness of some services/aspects across specific roles and in specific roles, respectively. While variation reveals how adaptable a service/aspect would be for integration into separate roles, uniqueness reveals how critical a service/aspect would be for the functioning of any application.

Table 1 Coupling of services/aspects to roles

		Role		
Service	**Aspect**	**Solo**	**Cloud support**	**Peer support**
Binding	Authentication	●	●	●
	Conformance			
Interaction	Performance	●	●	●
	Integrity			
	Collection	◇	◇	◇
	Extraction		○	
Unbinding	Deletion	●	●	●
	Notification		○	

Our literature review of mixing cloud computing and edge computing shows a strong embracement of Fig. 3a where edge nodes sit between data and cloud layers. Unfortunately, other forms of mix exist but are overlooked by the community. We demonstrated 3 additional forms considering applications’ requirements. For instance, an edge node could be the backbone of an application’s architecture without jeopardizing its functioning. And, an edge node could offer support to peers sustaining their performance and improving their reliability.

3.5 Implementation

To check the technical doability of the edges’ prescribed roles, we developed a testbed that would safeguard edge nodes from “themselves” and other third parties with respect to the demands that these nodes would issue/receive. Figure 5 is the testbed’s architecture featuring 4 modules, *policy manager*, *filter*, *monitor*, and *auditor*. In this architecture, *third party* refers to the data layer, edge layer, and cloud layer, *service provider* refers to the edge layer and cloud layer, and finally numbers refer to the chronology of interactions to ensure the definition of ODRL policies, the proper use of edge nodes, and the compliance of edge nodes' operations with these policies. The 4 modules are detailed below:

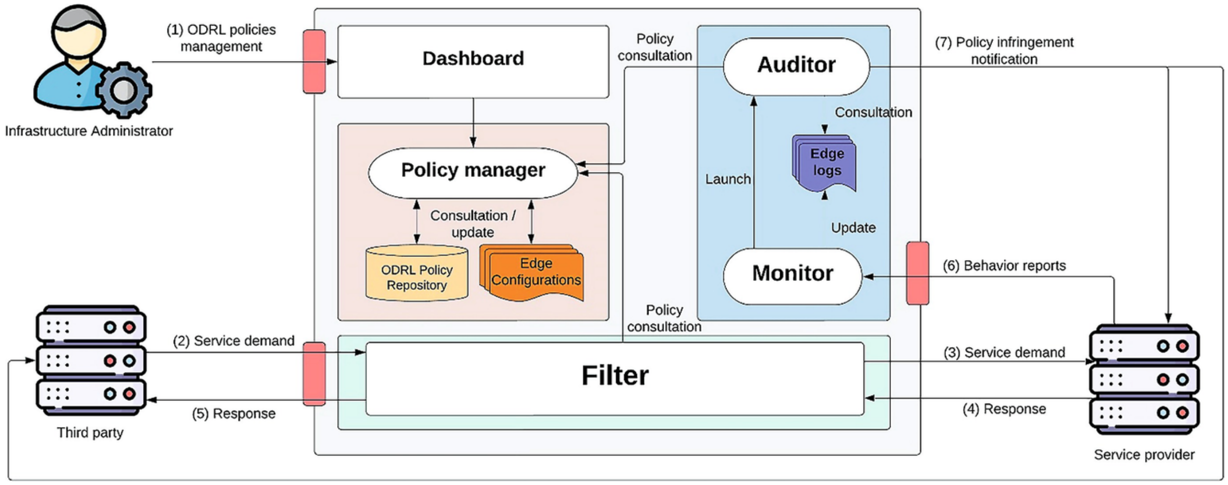

(1) ODRL policies management
Infrastructure Administrator
Dashboard
Policy consultation
Auditor
(7) Policy infringement notification
Consultation
Edge logs
Launch
Policy manager
Consultation / update
ODRL Policy Repository
Edge Configurations
Update
Monitor
(6) Behavior reports
Policy consultation
(2) Service demand
Filter
(3) Service demand
(5) Response
(4) Response
Third party
Service provider

Fig. 5 Testbed's architecture

The *policy manager* handles the ODRL policies that the *infrastructure administrator* provides through a *dashboard*. It organizes and stores the policies into *edge configuration* files. A file contains a specific edge's policies and is structured into different sections dedicated to binding, interaction, and unbinding services. This way of doing facilitates access and use of the policies by other modules.

The *filter* receives external demands from third parties and internal demands from the edge node itself regarding the potential uses of the edge node as per one of the roles that this edge node could take over. The filtering consists of associating demands with ODRL policies. Should a policy turn out verified, then the demand would be approved leading to invoking the relevant aspects included in the edge node's services so, such as transferring data from a data layer to an edge node and offloading data from an edge layer to an edge node. Otherwise, the demand is rejected.

The *monitor* tracks the operations that the edge node implements based on the internal/external demands it received and either permitted or obliged. Tracking outcomes are stored in the *edge logs* to be used by the *auditor* module at a later stage.

The *auditor* enforces the compliance of the edge node's operations with its owner's policies. To this end, the *auditor* resorts to both the ODRL policies and the *edge log* checking what has been permitted, prohibited, and obliged *versus* what has really happened. Depending on the nature of demands/operations that the edge node implemented, the *auditor* notifies the edge node's owner and/or third parties of any potential deviation of the edge node from its prescribed role.

Different tools and technologies were used for the development and deployment of the testbed: Python (www.python.org) to implement the modules, Flask (flask.palletsprojects.com/en/2.3.x.) and Bootstrap5 (getbootstrap.com) to create the dashboard interface for policy management by the administrator, JSON-LD to implement the *policies repository*, and, finally, YAML to represent *edge configuration* files. We deployed the testbed on the CEntre de Recherche en Information Scientifique et Technique in Algiers, Algeria (CERIST) cloud infrastructure using the open-source platform OpenStack (www.openstack.org). We used OpenStack's online dashboard "Horizon" to access CERIST's resources and create several compute instances, namely, Ubuntu-20.04 servers (virtual machines) where each server represents a node in the infrastructure. By launching these servers, the 3-layer infrastructure hosting the *data*, *edge*, and *cloud* nodes was created. The specification of each layer's nodes is reported in Table 2. We then deployed the testbed on another node of the infrastructure with 6 GB RAM, 250 GB HD and 2 CPUs in order to ensure service provisioning at the edge nodes with respect to both their prescribed roles and the defined ODRL policies. To comply with the definition of edge computing by being close to the data layer, all data nodes and edge nodes were deployed locally on the premises of CERIST's local network, while the cloud node was deployed on a remote network called Us_net located in the USA. SSH connections were used to transfer and launch the relevant Python scripts for each node.

Table 2 Technical configuration of the testbed

		Specification		
Layer	**# of nodes deployed**	**CPU(#)**	**RAM (GB)**	**HD (GB)**
Data	4	1	2	20
Edge	1	8	16	160
	2	4	8	80
	3	2	6	250
Cloud	1	8	32	80

For evaluation needs, our tests consisted of sending multiple service demands from and to an edge node and measuring the average response time for each demand. A testing service demand contains details like the requested service, action, target, as well as timestamps for sending, examining, and responding to the demand. Table 3 represents the list of service demands used for testing, while Fig. 6 represents the average response time per demand.

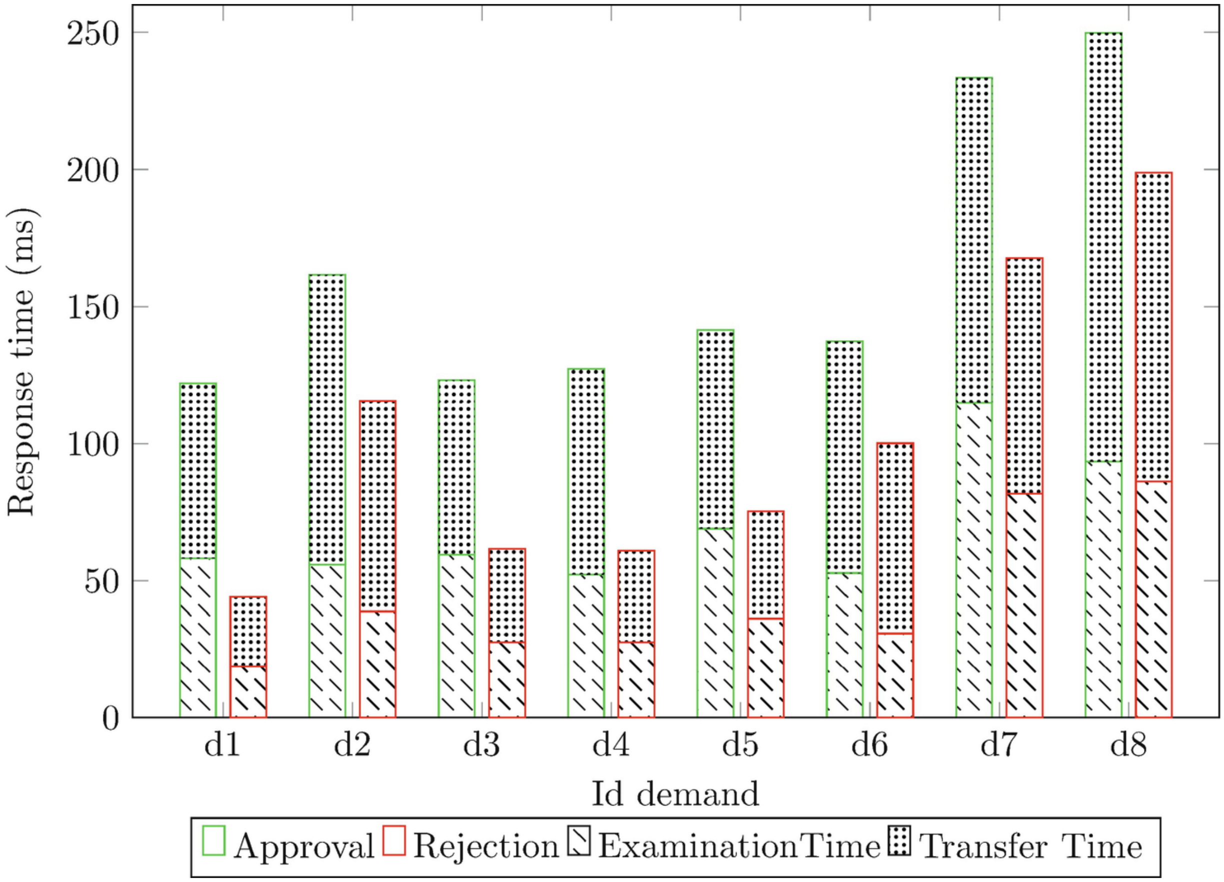


Fig. 6 Response time per service demand

Table 3 Testing service demands

Id	**Service demand’s type**
d1	Edge node annotation
d2	Binding to an edge node
d3	Data collection from a data node to an edge node
d4	Edge node data modification
d5	Collected data annotation

Id	Service demand’s type
d6	Data collection from an edge node to a cloud node
d7	Data collection from a cloud node to an edge node
d8	Data offloading to a peer edge

We note that the response time for each demand depends on whether the response is approved or rejected. When a response is approved, it generally takes longer to process compared to when it is rejected. This variation can be attributed to the verification of a higher number of policies and rules for approved demands. In the case of rejection, the *filter* immediately responds with a rejection upon encountering the first non-compliant policy, resulting in a relatively lower number of policies being verified. Moreover, the transfer time is extended for approved demands because a second transfer between the *filter* and service provider is required before issuing an approval, whereas a rejection response is directly transmitted to the third party that submitted the service demand.

We also explored the impact of varying the number of nodes on the testbed’s performance. We observed how the testbed’s response time is influenced by the scalability of the infrastructure. Using one single demand (d3 in Table 3), which concerns data collection from the data layer to the edge layer, we conducted several experiments, gradually increasing the number of policies granting access to different data nodes for the specified edge node, which reflected variations in the number of data nodes involved with the edge node. Figure 7 shows the response time to the same demand for different numbers of data nodes concerned by the edge node’s policies. We note a clear correlation between the number of involved data nodes in the policies and response time. Specifically, as the number of nodes increased, the response time also showed a corresponding increase. This finding suggests that the complexity of accessing and verifying more policies for one demand impacts the overall response time. Thus, it is important to consider this relationship to optimize the system and ensure better performance for large-scale infrastructures.

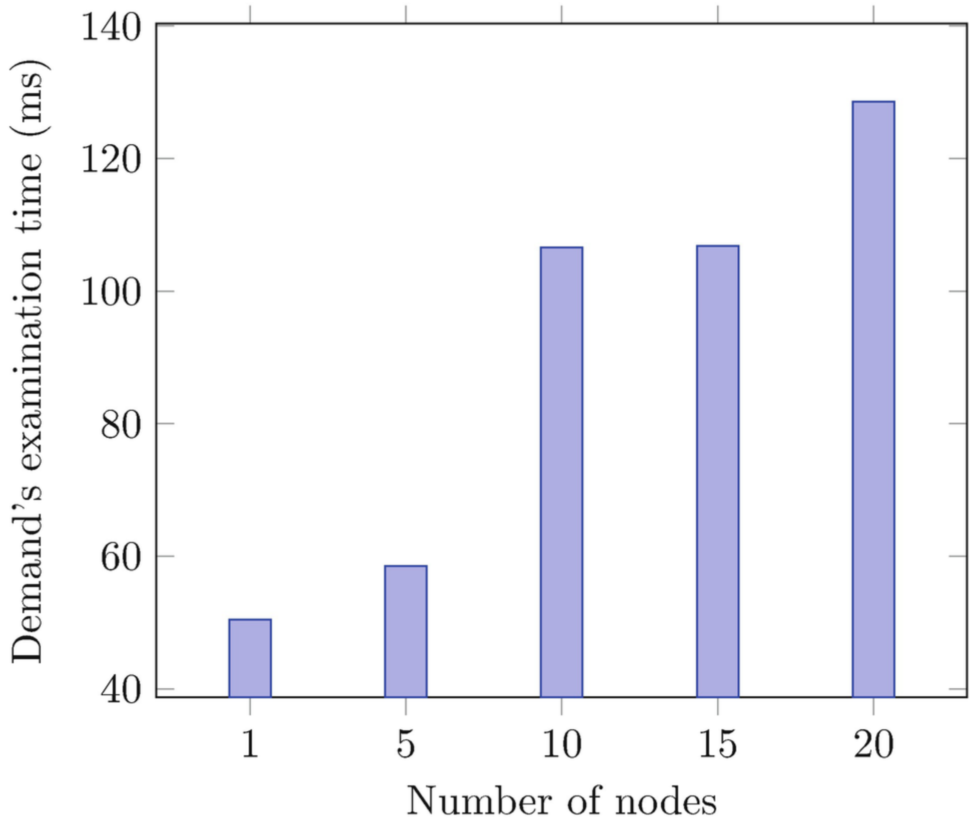


Fig. 7 Response time per number of nodes

4 Conclusion

In this chapter, we shed light on the collaboration that exists between cloud computing and edge computing. Indeed, some applications' requirements on data freshness and security make cloud computing inappropriate, and thus, edge computing becomes a strong alternative for these applications. Edge computing is to have processing, storage, and communication resources next to where data reside and/or collected preventing its long-distance transfer and unnecessary exposure. We exposed the multiple roles, solo, cloud support, and edge support that edge nodes could take over in enforcing this alternative. For each role, we specified the services that it should provision using a set of policies defined in ODRL. Some policies enforced data collection, while others enforced removal after unbinding. A testbed demonstrating the technical doability of edges' roles has been developed, tested, and evaluated with focus on edges' responses times. The testbed good complements some of the ongoing research on edge performance benchmark like those presented in [2]. In terms of future work, we would like to examine additional roles that edges could take over such as broker.

References

1. Al-Hasnawi, A., Carr, S., Gupta, A.: Fog-based local and remote policy enforcement for preserving data privacy in the Internet of Things. Internet Things **7**, 100069 (2019) [Crossref]
2. Blesson Varghese, B., Nan Wang, N., David Bermbach, D., Cheol-Ho Hong, C., Eyal de Lara, E., Weisong Shi, W., Christopher Stewart, C.: A survey on edge performance benchmarking. ACM Comput Surv **54**(3), 1–33 (2022)
3. Cano-Benito, J., Cimmino, A., García-Castro, R.: Injecting data into ODRL privacy policies dynamically with RDF mappings. In: Companion Proceedings of the ACM Web Conference 2023 (WWW'2023). Austin, TX, USA (2023)
4. Chen, Y., Deng, S., Zhao, H., He, Q., Li, Y., Gao, H.: Data-intensive application deployment at edge: a deep reinforcement learning approach. In: Proceedings of the IEEE International Conference on Web Services (ICWS'2019). Milan, Italy (2019)
5. De Donno, M., Tange, K., Dragoni, N.: Foundations and evolution of modern computing paradigms: cloud, IoT, edge, and fog. IEEE Access **7**, 150936–150948 (2019) [Crossref]
6. De Vos, M., Kirrane, S., Padget, J., Satoh, K.: ODRL policy modelling and compliance checking. In: Proceedings of the Third International Joint Conference on Rules and Reasoning (RuleML+RR'2019). Bolzano, Italy (2019)
7. Ghosh, A., Khalid, O., Bin Rais, R., Rehman, A., Malik, S., Ali Khan, I.: Data offloading in IoT environments: modeling, analysis, and verification. EURASIP J. Wirel. Commun. Netw. **2019**, 53 (2019)
8. Hicks, K.: The 5 Most Important Smart Tech Devices for Senior Safety (2019). https://www.aplaceformom.com/caregiver-resources/articles/smart-tech-devices-for-senior-safety
9. Khebbeb, K., Hameurlain, N., Belala, F.: A Maude-based rewriting approach to model and verify cloud/fog self-adaptation and orchestration. J. Syst. Arch. **110**, 101821 (2020) [Crossref]
10. Li, C., Tang, J., Tang, H., Luo, Y.: Collaborative cache allocation and task scheduling for data-intensive applications in edge computing environment. Future Gener. Comput. Syst. **95**, 249–264 (2019) [Crossref]
11. Maamar, Z., Baker, T., Faci, N., Ugljanin, E., Al-Khafajiy, M., Burégio, V.: Towards a seamless coordination of cloud and fog: illustration through the Internet-of-Things. In: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing (SAC'2019). Limassol, Cyprus (2019)

12. Maamar, Z., Faci, N., Yahya, F., Kajan, E.: Trust-empowered, IoT-driven legitimate data offloading. In: Proceedings of the Workshops of EDBT/ICDT’2021 Joint Conference. Nicosia, Cyprus (online) (2021)
13. Masip-Bruin, X., Marín-Tordera, E., Tashakor, G., Jukan, A., Ren, G.: Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud computing systems. IEEE Wirel. Commun. **23**(5), 120–128 (2016) [Crossref]
14. Peter, M., Timothy, G.: The NIST definition of cloud computing. Technical Report 800-145, National Institute of Standards and Technology (NIST) (2011)
15. Puliafito, C., Mingozzi, E., Longo, F., Puliafito, A., Rana, O.: Fog computing for the Internet of Things: a survey. ACM Trans. Internet Technol. **19**(2), 1–41 (2019) [Crossref]
16. Rausch, T., Rashed, A., Dustdar, S.: Optimized container scheduling for data-intensive serverless edge computing. Future Gener. Comput. Syst. **114**, 259–271 (2021) [Crossref]
17. Revel, M., Lamercerie, A., Foret, A., Miklós, Z.: Extracting ODRL digital right representations from license texts using AMR. In: Proceedings of the 6th Workshop on Automated Semantic Analysis of Information in Legal Text co-located with the 19th International Conference on Artificial Intelligence and Law (ICAIL’2023). Braga, Portugal (2023)
18. Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L.: Edge computing: vision and challenges. IEEE Internet Things J. **3**(5), 637–646 (2016) [Crossref]
19. Singh, S.: Optimize cloud computations using edge computing. In: Proceedings of the IEEE International Conference on Big Data, IoT and Data Science (BID’2017). Pune, India (2017)
20. Varghese, B., Wang, N., Nikolopoulos, D., Buyya, R.: Feasibility of Fog Computing (2017). CoRR arXiv preprint arXiv:1701.05451 abs/1701.05451
21. W3C: ODRL Information Model 2.2 (2018). https://www.w3.org/TR/2018/REC-odrl-model-20180215/, (Visited January 2023)
22. Yahya, F., Maamar, Z., Boukadi, K.: A multi-criteria decision making approach for cloud-fog coordination. In: Proceedings of the 34th International Conference on Advanced Information Networking and Applications (AINA’2020). Caserta, Italy (online) (2020)
23. Zada Khan, W., Ahmed, E., Hakak, S., Yaqoob, H., Ahmed, A.: Edge computing: a survey. Future Gener. Comput. Syst. **97**, 219–235 (2019) [Crossref]
24. Zhang, Z., Cho, M., Wang, C., Hsu, C., Chen, C., Shieh, S.: IoT security: ongoing challenges and research opportunities. In: Proceedings of the IEEE 7th International Conference on Service-Oriented Computing and Applications (SOCA’2014). Matsue, Japan (2014)

Footnotes

[1] Puliafito et al. report that “*the average round trip time between an Amazon Cloud server in Virginia (U.S.A.) and a device in the U.S. Pacific Coast is 66ms; it is equal to 125ms if the end device is in Italy and reaches 302ms when the device is in Beijing*” [15].

[2] In this chapter, edge and fog are treated the same.

[3] www.statista.com/topics/6173/edge-computing/$sharp$dossierKeyfigures.

[4] animelandmakiba.com/2020/12/computer-societys-top-10-technology-trends-in-2021.

OceanofPDF.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
S. Pal et al. (eds.), *IoT Edge Intelligence*, Internet of Things: Technology, Communications and Computing
https://doi.org/10.1007/978-3-031-58388-9_6

Unsupervised Time Series Anomaly Detection for Edge Computing Applications: A Review

Danlei Li[1] ✉, Nirmal Nair[1] ✉ and Kevin I-Kai Wang[1] ✉
(1) Department of Electrical, Computer, and Software Engineering, The University of Auckland, Auckland, New Zealand

✉ **Danlei Li (Corresponding author)**
Email: dli459@aucklanduni.ac.nz

✉ **Nirmal Nair**
Email: n.nair@auckland.ac.nz

✉ **Kevin I-Kai Wang**
Email: kevin.wang@auckland.ac.nz

Keywords Edge computing – Anomaly detection – Sensor data quality – Unsupervised learning – Concept drift

---

1 Introduction

The Internet of Things (IoT) is a group of interconnected devices that can acquire and exchange data from various sources, including multiple types of sensors and large machines. In the IoT field, edge computing is a paradigm that enables data processing to occur closer to the data source. This aids in minimizing latency and providing responses in real time. Good data quality is of critical importance to the trained model quality in IoT and edge computing applications, which can lead to better decision-making, operational efficiency, and user satisfaction. However, data quality can be significantly impacted by anomalies (sometimes also referred to as outliers), which are data instances that differ from the original data pattern or distribution. Subsequently, the accuracy of edge computing models will also be affected. At the same time, IoT-based sensor systems are commonly large scale and widely distributed, and manual inspection of each sensor is impractical. To ensure the long-term data quality, it is important to identify these anomalies using robust and long-term reliable anomaly detection approaches. In addition, due to the scale and wide distribution, it is also challenging to label in situ sensor data for the purpose of training, and hence the majority of anomaly detection methods are unsupervised. Therefore, the goal of this chapter is to examine the state-of-the-art unsupervised anomaly detection methods for edge computing applications and investigate their potential to achieve long-term reliability for time series data.

Anomaly detection methods have been initially investigated in the domains of image mining and computer vision. Existing research indicates that statistical, machine learning, and deep learning algorithms are the primary categories of anomaly detection methods [12, 20, 60]. A statistical model is a mathematical instrument used to approximate reality. It relies primarily on quantitative analysis, summarizes empirical principles derived from earlier data, and infers whether a future instance is an anomaly. Machine learning enables the models to autonomously recognize patterns through classification, regression, or clustering. Then, predictions are made based on the learned pattern. In anomaly detection problems, it will learn from the observed normal instances and identify points that deviate from the expected behavior. Deep learning pertains to the machine learning family. Artificial neural networks and representation learning are utilized in deep learning methods. It employs neurons to simulate how humans acquire certain types of knowledge to identify more complex patterns in the data. In contrast to image-based anomalies, the study of time series anomalies has recently acquired popularity due to the proliferation of electronic devices and sensor applications in the IoT area [25]. Sensor data instances are recorded sequentially and correlated in time. These instances constitute the time series. Currently, most proposed solutions for time series data still employ image anomaly detection algorithms [21, 25, 46].

Since these existing algorithms were not designed for time series, they cannot utilize valuable temporal information and analyze context associations between data. Therefore, temporal information is very often ignored in many existing anomaly detection frameworks. In addition, it is a common assumption in statistics, machine learning, and deep learning that data is IID (independent and identically distributed). However, the underlying distribution of data usually changes due to natural variations of sensor behaviors and physical environments in time series data. The changes in data over time are referred to as concept drift. They are rarely considered or addressed in existing anomaly detection methods. The concept drifts could be caused by different factors such as changes in external environment or degradation in internal sensor mechanism. Therefore, concept drifts have a substantial effect on the performance of trained models over the long term. This issue is further amplified by the fact that many time series models are only trained with few initial instances for typical IoT and edge computing applications. Consequently, ensuring the long-term reliability of anomaly detection models is overlooked in edge computing applications. Concerning concept drift, there exists a variety of techniques for detecting concept drift. In order to ensure long-term reliability, the unsupervised anomaly detection method should have the ability to adapt to concept drifts.

In addition to the long-term reliability, the anomaly detection model must start as soon as possible to aid in data quality evaluation in edge computing scenarios. Even during the initial stage of system startup, the model should be able to produce an acceptable detection based on a small quantity of observed data. Therefore, the ideal model should require only few training instances. Our vision of the architecture of long-term anomaly detection framework is shown in Fig. 1. The initial model will be trained with a small training set and subsequently updated based on the concept drift detection result. Our objective in this chapter is to review existing unsupervised anomaly detection methods and investigate their potential to be integrated with concept drift detection to achieve reliable and unsupervised anomaly detection for long-term time series. Therefore, we will summarize and analyze the state-of-the-art techniques of unsupervised anomaly detection and concept drift detection in Sects. 2 and 3.

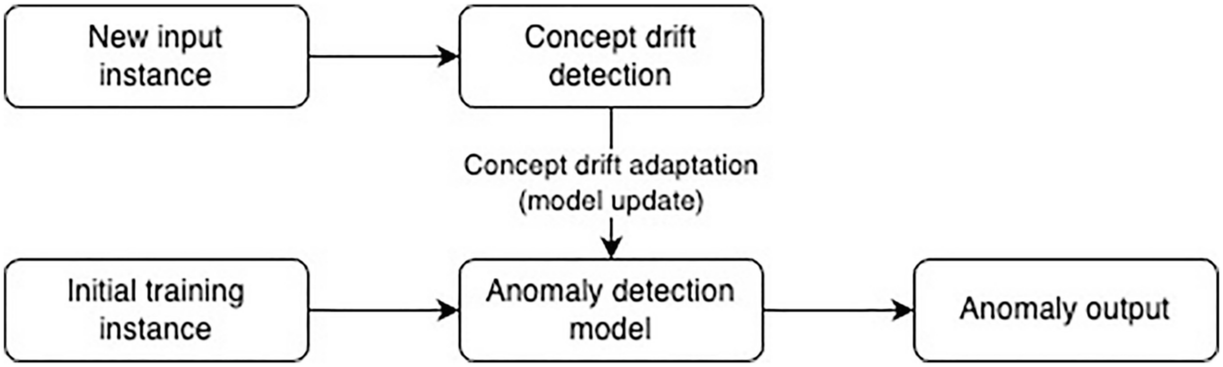

New input instance
Concept drift detection
Concept drift adaptation (model update)
Initial training instance
Anomaly detection model
Anomaly output

Fig. 1 Long-term anomaly detection framework architecture

The proposed framework encourages the investigation of anomaly detection with concept drift detection techniques. As shown in Fig. 2, the remainder of this chapter is structured as follows. Section 2 analyzes and categorizes unsupervised time series anomaly detection methods. Section 3 summarizes concept drift detection methods. Section 4 does further analysis on the existing anomaly detection methods in terms of their long-term reliability and ability to adapt to concept drift, and hence pointing out the challenges and promising future directions. At the end, Sect. 5 concludes this chapter.

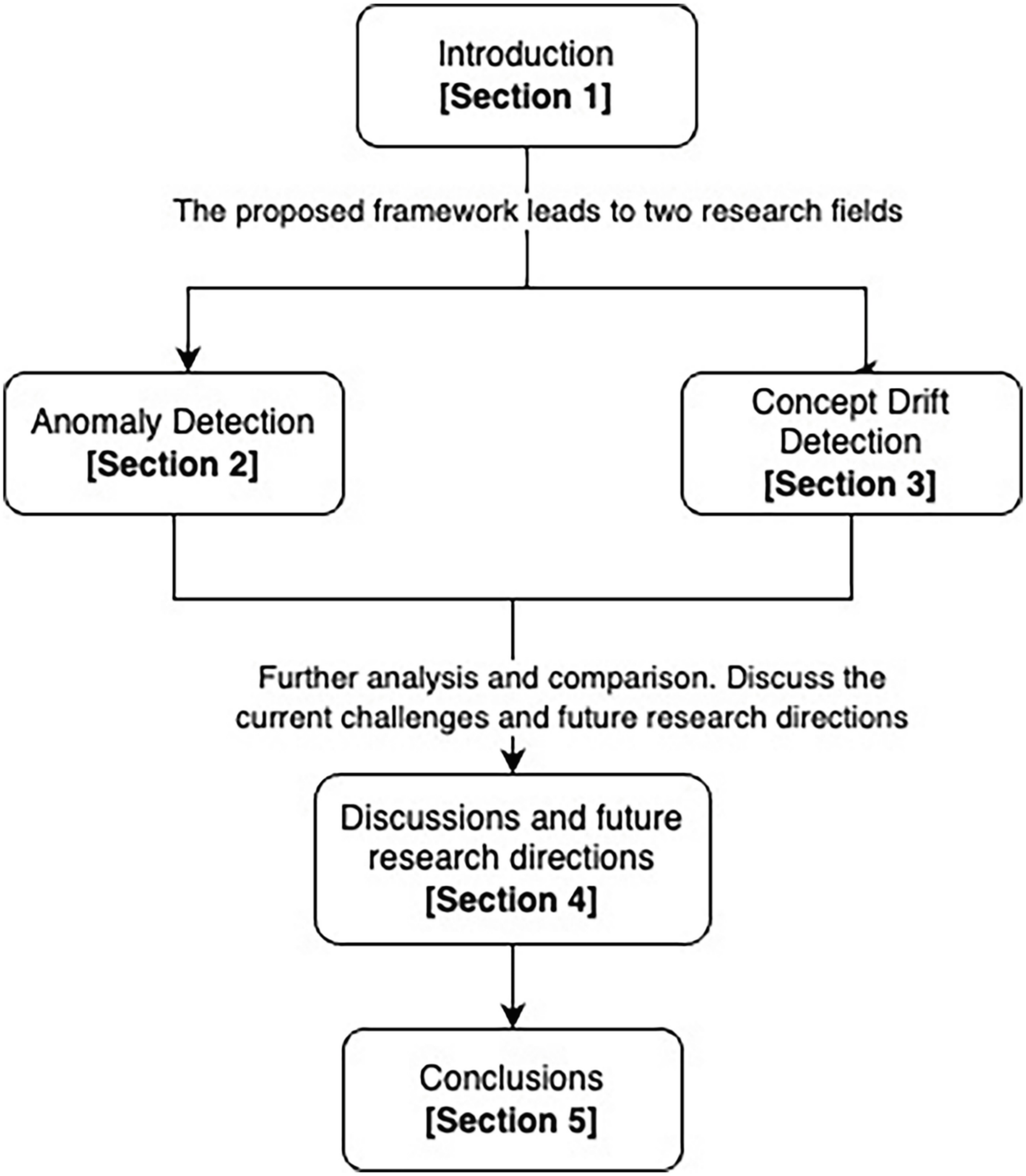

Introduction
[Section 1]
The proposed framework leads to two research fields
Anomaly Detection
[Section 2]
Concept Drift Detection
[Section 3]
Further analysis and comparison. Discuss the current challenges and future research directions
Discussions and future research directions
[Section 4]
Conclusions
[Section 5]

Fig. 2 Structure and flow of the chapter

2 Anomaly Detection

In recent studies, anomalies in time series [25] are generally divided into the following three categories:

- Point Anomaly: A single data instance is regarded as unnatural compared to the other instances. The point anomaly occurs most frequently and is the most apparent.
- Contextual Anomaly: This is a data instance that falls within a normal range but is regarded as abnormal by combining contextual and behavioral features.
- Collective Anomaly: A collection of continuous instances deviates from the overall dataset. Single instances in the collection may not be out of the ordinary, but their existence as a group is abnormal.

Figure 3 shows the categories of the unsupervised anomaly detection methods that target time series data. In the following subsections, we analyze thoroughly each category of anomaly detection algorithms. We explain the operating assumptions and introduce the fundamental techniques in each category. After that, we analyze a selection of the state-of-the-art algorithms commonly employed in each category and discuss their advantages and disadvantages. Finally, the overall pros and cons of each method category and the corresponding connection with concept drift are discussed.

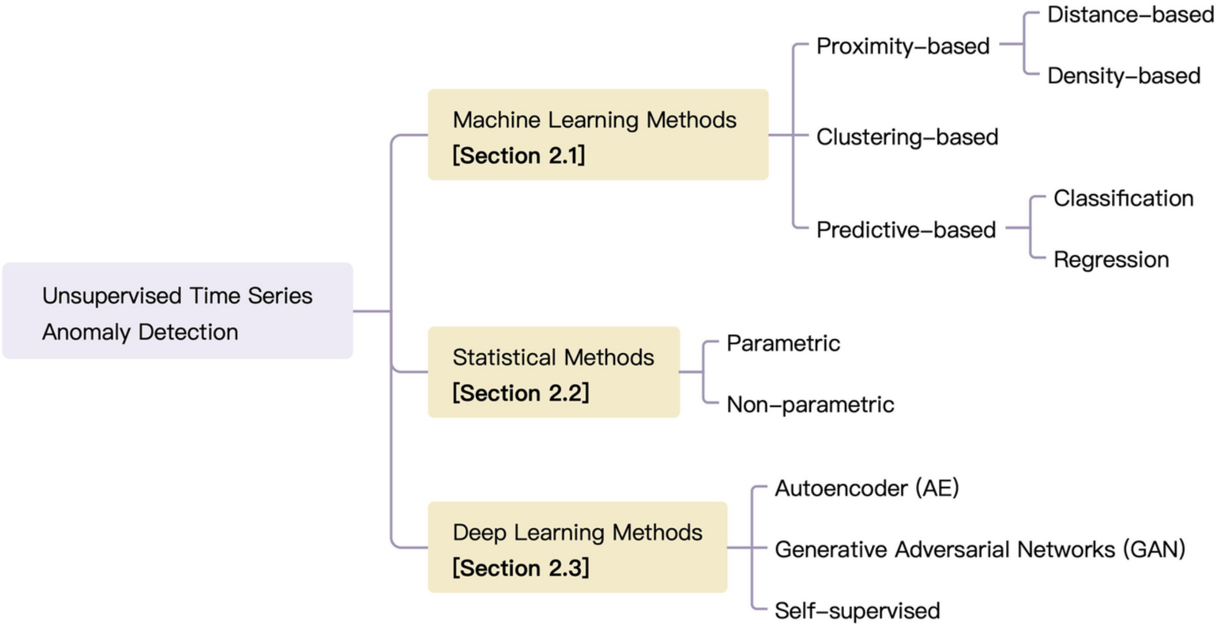

Unsupervised Time Series Anomaly Detection
Machine Learning Methods [Section 2.1]
Proximity–based
Distance–based
Density–based
Clustering–based
Predictive–based
Classification
Regression
Statistical Methods [Section 2.2]
Parametric
Non–parametric
Deep Learning Methods [Section 2.3]
Autoencoder (AE)
Generative Adversarial Networks (GAN)
Self–supervised

Fig. 3 Categories of unsupervised anomaly detection methods in time series data

2.1 Machine Learning Methods

As shown in Fig. 3, the main categories of machine learning methods include proximity-based, clustering-based, and predictive-based algorithms. Different criteria are used to differentiate anomalies in different algorithms.

2.1.1 Proximity-Based

The methods in proximity-based category [1] assume that anomaly instances typically occur in sparse regions. Therefore, they commonly maintain a distance from their neighbors. In contrast, normal instances occur in concentrated regions, and hence they should be close to their neighbors. This type of algorithms can solve anomaly detection problems without labels. Therefore, proximity-based algorithms are generally considered as unsupervised techniques in the anomaly detection area. All proximity-based methods aim to estimate the anomaly score according to the distance between instances and their $k\mathrm{th}$ nearest neighbors or relative density. According to how the neighbors are chosen, these algorithms are further subdivided into distance-based and density-based methods.

Distance-based method An anomaly score in distance-based methods is typically computed as the distance between data instance and its neighbors. Data instances with high anomaly scores are further away from their neighbors and hence are identified as anomalies. Distances are typically measured by the Manhattan distance ($L1$) or the Euclidean distance ($L2$). The most representative distance-based method is the $k\mathrm{th}$ nearest neighbor (*k*NN). It calculates anomaly scores based on the distance between each data instance and its $k\mathrm{th}$ nearest neighbor.

***k*NN methods** The classic *k*NN method is a supervised method. It assumes that similar instances exist nearby. The $k\mathrm{th}$ nearest neighbor region of each instance $\mathbf{q}$ is denoted by $NR_k(\mathbf{q})$, which includes a set of points that lies in or on the hypersphere of radius *k* distance and centered with $\mathbf{q}$. The $k\mathrm{th}$ nearest neighbor is the instance that lies on the hypersphere boundary, denoted by $N_k(\mathbf{q})$. Regarding the nearest neighbor measurement, several formulas exist to estimate the distance between each instance and its neighbors. In addition to the $L1$ and $L2$ distance, Minkowski and Chebyshev distances are occasionally applied in *k*NN measurement. The *kNN* takes an unsupervised approach to anomaly detection. Instead of classifying the instances depending on the labels of their neighborhood, the *k*NN-based anomaly detection methods calculate the anomaly scores according to the distance between the data point $\mathbf{q}$ ($\mathbf{q} \in \mathbb{X}$) and its $k\mathrm{th}$ nearest neighbor [5].

Although *k*NN is simple, the main problem is selecting the optimal *k* value for any particular dataset since the ground truth is unavailable in unsupervised problems. Standard model optimization methods such as cross-validation and grid search cannot help with the selection of *k*. To solve this problem, the average *k*-nearest neighbor detector (average *k*NN) is proposed [4], which uses a more robust way to choose neighbors.

In addition, other variant algorithms such as Angle-Based Outlier Detection (ABOD) and FastABOD [45] have been proposed to use angles to compute the anomaly score. The LeSiNN [56, 61] identifies the least similar nearest neighbors and reduces time complexity by using ensemble and subsampling techniques. AnomalyDetect [69] is a recent online distance-based methods which has an online update strategy using sliding window techniques. Langfu et al. [47] applies the Dynamic Time Warping (Fast-DTW) method to oversample the time series, accelerating the method's efficiency. However, experimental results still demonstrate that the model output is highly sensitive to the *k* value.

Density-based These methods group data instances according to their distances from their corresponding group. Normal instances are located in high-density regions. As a commonly employed algorithm Local Outlier Factor (LOF) [14] is designed for anomaly detection problems.

LOF Similar to the *k*NN method, the two key parameters in density-based methods are the $k\mathrm{th}$ distance and the $k\mathrm{th}$ nearest neighbor region ($NR_k(\mathbf{q})$). Since the $k\mathrm{th}$ distance is equivalent to the radius of the $k\mathrm{th}$ nearest neighbor hypersphere, the instances in the sparser region would have the longer $k\mathrm{th}$ distance. The density-based techniques use the inverse of $k\mathrm{th}$ distance to estimate the density of the corresponding instance. The LOF calculates the anomaly score according to each instance's local density and the average local density ratio of their $k\mathrm{th}$ nearest neighbors. It has the following three steps to compute the final result. Assume we have a data instance $\mathbf{q} \in \mathbb{X}$. The first step is to find the $k\mathrm{th}$ distance value. It is denoted by $d(\mathbf{q}, \mathbf{o})$, where $\mathbf{o}$ is the $k\mathrm{th}$ neighbor of $\mathbf{q}$. After that, the second step computes the local reachability density (LRD). The LRD value of an instance $\mathbf{q}$ is the inverse of its corresponding $Average\ RD_{\mathbf{q}}$ as in (1):

$$LRD(\mathbf{q}) = \frac{1}{Average\ RD_{\mathbf{q}}} = \frac{1}{\frac{1}{k}\sum_{n=1}^{k}\max(k\text{th}\ distance(\mathbf{o}_n), \|\mathbf{q}-\mathbf{o}_n\|)}, \tag{1}$$

where $\{\mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_k\} \in NR_k(\mathbf{q})$, $k\text{th}\ distance(\mathbf{o})$ is the $k\text{th}$ distance of $\mathbf{o}$, and $\|\mathbf{q}-\mathbf{o}\|$ is the distance of $\mathbf{q}$ and $\mathbf{o}$. In the last step, the LOF score is calculated as the ratio of the local density of each instance and the average local density of the instances in its $k\text{th}$ nearest neighbor region as (2)

$$LOF_k(\mathbf{q}) = \frac{\sum_{\mathbf{o}\in NR_k(\mathbf{q})} LRD_k(\mathbf{o})}{|NR(\mathbf{q})|.LRD_k(\mathbf{q})}. \tag{2}$$

This quantity can accurately measure the density difference between the instances and their neighbors. If $LOF_k(\mathbf{q})$ is close to 1, the density of this instance is similar to its neighbors within $k\text{th}$ distance. If the value is much larger than 1, it indicates that this instance is in a dense area. Finally, if the value is much smaller than 1, then the density of this instance is less than other points in its neighborhood. In this case, the instance will be identified as an anomaly.

The connectivity-based outlier factor (COF) [73] can be considered as a variant of LOF method that calculates the neighborhood of instances in an incremental mode. The distinction is their $k\text{th}$ nearest region selection strategy. First, choose the closest instance of the input instance, and add it to the neighborhood. COF will then keep adding new neighbors to the neighborhood. The selected instance should be closest to all existing instances in the current neighborhood (not just closest to the input).

Density-based methods usually have high time complexity because of the pair-wise distance estimation for density calculation. FastLOF [30] has been proposed to improve the efficiency of LOF. The LiNearN [77] has a linear time complexity. However, it is still sensitive to the parameter *k*. Incremental LOF [64] and DILOF [59] are two online update methods that are designed for continuous data stream. However, they are sensitive to the window size. In addition, incremental LOF requires enormous memory space. The benefits and drawbacks of the proximity-based algorithms are summarized in Table 1.

Table 1 Proximity-based anomaly detection methods

Algorithm	Advantages	Disadvantages
Distance-based		
KNN [72]	Easy to implement	Sensitive to the parameters and nearest neighbor selection
		High time complexity
Average KNN [4]	More robust on choosing neighbors	High time complexity
ABOD/FastABOD [45]	Free of parameters	Mediocre performance on one-dimensional data or univariate time series
	Supports high-dimensional data	High time complexity
LeSiNN [56, 61]	Achieves linear time complexity	Requires a large training set
	Supports high-dimensional data	Rely on multiple new parameters
	Works on the large dataset	Difficult to control model performance
AnomalyDetect [39]	Online update	Sensitive to multiple parameters such as k, sliding window size, and the anomaly threshold value
	Has a certain degree of adaptability to new data	
	Reduces the time complexity by using the time–space trade-off mechanism	
Langfu et al. [47]	Reduce the time complexity by using the DTW over-sampling method	Sensitive to the k selection
Density-based		
LOF [14]	Simple and designed for anomaly detection problem	Needs the pair-wise distance to estimate the density, which causes the high time complexity
COF [73]	More robust of choosing neighbors	High time complexity
FastLOF [30]	Improves the efficiency of LOF	Requires a large training set
	Works on the large dataset	
LiNearN [77]	Linear time complexity	Sensitive to the parameters and nearest neighbor selection
	Can handle high-dimensional data	

Algorithm	**Advantages**	**Disadvantages**
Incremental LOF [64]/DILOF [59]	Online update	Incremental LOF requires huge memory space
	Has some adaptability to new data	Sensitive to the window size

In conclusion, the majority of proximity-based anomaly detection methods have high time complexity because of the need to compute pair-wise distance detection. In addition, they are always sensitive to parameters, such as the *k* value and the window size in online models. Regarding time series data, most of them do not consider time information in the data during the detection process, instead of relying solely on the distance to determine the anomaly. Due to these weaknesses, their long-term performance stability is typically unstable. Few state-of-the-art models, which include AnomalyDetect, Incremental LOF, and DILOF, involve online updated strategies. However, their update strategies do not consider concept drift or its potential impact on the model. Therefore, they still cannot avoid the effect of concept drift.

2.1.2 Clustering-Based

These methods can be considered as an extension of density-based methods and create clusters based on data density. The central assumption of them is that anomalies are commonly from smaller or sparser clusters, while normal instances are from bigger or denser clusters [21]. The data instances that are closer to or within dense clusters are considered as normal points. The data instances that are further away from these dense clusters or located in the sparse clusters are labeled as anomalous. Clustering-based algorithms do not require labels to train the model. Therefore, these algorithms are generally considered as unsupervised learning methods.

As a traditional clustering method, *K*-means [43] can be applied to anomaly detection by adding a threshold value. Data instances will be identified as anomalies if the distance between them and their nearest centroid exceeds the threshold value. Another popular algorithm, the clustering-based local outlier factor (CBLOF) is proposed in 2013 [36]. For each data instance, the anomaly score will be calculated by considering its distance from its corresponding centroid and the size of the cluster.

***K*-means** This method assumes that anomalous data instances stay far away from their corresponding centroid [21]. Cluster-based anomaly detection algorithms generally have two steps to work out the anomaly score. The first step is to cluster each data instance by applying existing clustering methods. After that, the distance between the instance $\mathbf{q}$ and the cluster center $c(\mathbf{q})$ will be used as anomaly scores as shown in (3):

$$S = d(\mathbf{q}, c(\mathbf{q})), \tag{3}$$

where *S* represents the anomaly score.

CBLOF This method can be considered as a combination of clustering and LOF. It applies the clustering methods first. The anomaly score for each instance will be affected by its corresponding cluster size and its distance from the centroid. Small clusters are possible to be identified as abnormal clusters. In this case, anomalies in the abnormal cluster will also be closer to their centroid. Therefore, we also need to define the type of clusters (large or small). We can sort the clusters based on their sizes and select a threshold to define the large cluster and small cluster. If an instance is from a larger cluster, we directly calculate the distance of this instance to its cluster centroid. If an instance belongs to a smaller cluster, we must calculate the distances between the instance and the centroids of large clusters and choose the minimum distance value.

Due to the fact that most clustering-based methods can only consider spherical clusters, Novel Local Outlier Detection (NLOD) [27] has been proposed to deal with arbitrary-shaped clusters for a large dataset. However, NLOD assumes that the distances between cluster centers and other instances within the corresponding cluster are relatively great. This assumption cannot always hold in many datasets. Li et al. [51] generate a method by applying fuzzy clustering and sliding window technique. This method is effective on multivariate time series but requires a large training set and has high computational cost and time complexity. The benefits and drawbacks of the above methods are summarized in Table 2.

Table 2 Clustering-based anomaly detection methods

Algorithm	Advantages	Disadvantages
K-mean [43]	Simple and easy to implement	Only consider spherical clusters

Algorithm	Advantages	Disadvantages
	High accuracy	Sensitive to predefined cluster number
	Application agnostic	
CBLOF [36]	Considers both density and size of the cluster	Contains and is sensitive to multiple parameters
NLOD [27]	Works on arbitrary-shaped clusters	Has specific assumption of cluster centers which are not always true
Li et al. [51]	Effective on multivariate time series	High computational cost
	Application agnostic	High time complexity
		Require a large training set

Typically, clustering-based algorithms are highly influenced by parameters, like the distance thresholds and the cluster numbers. Because the instances lack labels, it is challenging to define these values. Clustering methods also require an extensive amount of data to ensure a complete data distribution representation. When we only have a small amount of training data in edge computing applications, clustering-based methods are likely to fail to maintain long-term stable performance. Also, clustering-based methods require the dataset to have obvious and stable clustering boundary. If concept drift occurs and alters the clustering boundaries, the accuracy of the model will decrease drastically.

2.1.3 Predictive-Based

On the basis of the variable type, the predictive-based methods can be divided into two separate groups. Classification-based algorithms are designed for categorical variables, whereas regression-based algorithms are for continuous variables.

Classification Classification-based algorithms aim to learn from the provided features and and identify the boundary between the normal and abnormal classes [21]. The data instances that fall outside the boundary will be considered as abnormal [3]. Among the unsupervised classification methods, one-class support vector machine (OCSVM) [70] is a representative one. However, the performance of classic classification technique might be very poor since normal instances and anomalies are often imbalanced in the dataset. On the other hand, the isolation forest (iForest) method has been proposed to isolate the anomalies from the normal instances. iForest is also sometimes considered a clustering-based method.

OCSVM The traditional SVM is a supervised technique for classification problems. The value margin is the distance between the decision hyperplane and its nearest instances. Assume $f(\mathbf{q}_i) = w^T(\mathbf{q}_i) + b$ is the decision hyperplane, where *b* is a constant number, *w* is a normal vector, and $\mathbf{q}_i$ is a data instance with label Y_i (Y_i equals to 1 or -1). The classifier works as follows: If $w^T(\mathbf{q}_i + b) \geq 0$, then $Y_i = 1$, and if $w^T(\mathbf{q}_i + b) < 0$, then $Y_i = -1$. The SVM model aims to optimize the decision boundary to maximize the margin as in (4):

$$\arg\max_{w,b}\{\frac{1}{\|w\|}\min_i[Y_i.(w^T(\mathbf{q}_i) + b)]\}. \tag{4}$$

Based on this basic knowledge of the SVM model, the OCSVM is an unsupervised model used in anomaly detection problems. This method is popular and widely used in IoT-related fields [79]. It assumes that all the normal data are from one class, while anomalies are not. The OCSVM model will only learn one normal data class. For each instance, if it falls within the learned region, then the instance will be reported as normal. Otherwise, it will be declared as an anomaly. Due to the data imbalance between the normal instances and anomalies, the OCSVM performance is sometimes very poor.

Temporal Hierarchical One Class (THOC) [71] has been proposed by applying the recurrent neural network (RNN). It has more competitive accuracy but requires a large training set and high time complexity. Another popular algorithm is Deep Support Vector Data Description (Deep SVDD) [67]. This method uses a deep neural network to compute the data-enclosing hypersphere and constructs a hypersphere ball to represent data normality. All instances outside the ball are identified as anomalies. However, this approach is generally applied to image data and high-dimensional datasets, which is out of this chapter's scope.

iForest This is a popular ensemble method with superior performance on many real-world datasets. The underlying assumption of iForest is anomalies should occur very rarely and are easy to isolate from normal instances. To make the model more robust, iForest consists of a group of iTree (i.e., the base detectors). During the training phase, each base detector is constructed by building a binary tree. A feature and a splitter value will be selected randomly at each level. We keep partitioning the training data into two nodes based on the splitter value. In the testing phase, each testing instance q will pass through each base detector, and the depth of the leaf node containing the testing instance will be computed. Since anomalies should be easily separated, their depth should be smaller than in normal instances. The average depth across all base detectors is used as the final anomaly score.

The iForest achieves high accuracy and low time complexity in both the training and testing phases. However, when there are too many anomalies in a dataset, it is challenging to identify them using the isolation method. The recent method, Extended iForest [35], further improved the accuracy of the iForest using the random hyperplane cuts technique. Carletti et al. [16] proposed an interpretability method which facilitates feature selection in iForest, thereby enhancing the performance of the iForest model. Nonetheless, the experiments still indicate that the model performs poorly when the dataset contains a significant number of anomalies.

2.1.4 Regression

In some survey articles, regression models are also considered as parametric statistical methods. By learning the long-term trends, the regression could predict the values of subsequent data instances. If a new instance deviates significantly from the expected value, it is deemed anomalous. Typical regression techniques, such as the linear regression technique, can also be easily used in anomaly detection problem.

In regression techniques, the residual value is used to assess the difference within the actual and predicted values. Typically, the residuals derived from the process of model fitting are analyzed to determine which instances deviate from the fitted regression model. The larger residual value means a higher anomaly score. The traditional regression-based anomaly detection methods typically contain the following two steps.

- Build a regression model over the training dataset
- Calculate the residual of each test instance with the following equation. This value will be used as the anomaly score.

$$S = residual = y_i - \hat{y}_i, \tag{5}$$

where y_i is the value of the actual test instance and $\hat{y}_i$ is the value predicted by the regression model.

An Auto-Regressive Moving Average (ARMA) model [22] has been proposed to detect anomalies on stationary time series and was later extended to Auto-Regressive Integrated Moving Average (ARIMA) [9].

ARMA This algorithm can comprehensively express the relationship between the latest instance and historical data. An ARMA model will use the last *p* instances and *q* errors to complete the prediction:

$$\hat{\mathbf{q}}_t = \epsilon_t + \sum_{i=1}^{p} \varphi_i \mathbf{q}_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i}. \tag{6}$$

In (6), $\hat{\mathbf{q}}_t$ is the estimated value of $\mathbf{q}_t$. $\{\mathbf{q}_{t-1}, \ldots, \mathbf{q}_{t-p}\}$ are the last *p* observations, and *p* also represents the order of the relationship equation. The elements in set $\{\varphi_1, \ldots, \varphi_p\}$ are the constants learned from the data and can be understood as the self-relation coefficients. This set describes the probability of a correlation of the current value and predicted result. The values ϵ_t and ϵ_{t-i} are the prediction errors for the two periods, and the elements in set $\{\theta_1, \ldots, \theta_q\}$ are the coefficients of the prediction error.

The ARMA model was originally used to process univariate time series. It can be used in multivariate data by linearly combining several univariate time series models. An important assumption in the ARMA models is that the time series should be stationary. This means the time series follows its existing pattern captured by the regression model with no significant change in mean and variance. Nonetheless, the fact that datasets are frequently non-stationary is one of the most significant issues. As a result, the ARIMA model has been proposed.

ARIMA It can be considered as an extension of the ARMA model. By utilizing integrated methodologies, ARIMA can be used in weak stationary time series. Integrated refers to the differentiation of unprocessed instances in order to stabilize the time series. The degree of difference is denoted as *d*. ARIMA will build an $ARMA(p, q)$ model over the $d\text{th}$-order stationary series, denoted as $ARIMA(p, d, q)$ with the following equation:

$$\hat{\mathbf{q}}_t^{*d} = \epsilon_t + \sum_{i=1}^{p} \varphi_i \mathbf{q}_{t-i}^{*d} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i}, \tag{7}$$

where *d* is the order of differences needed for becoming stationary.

The Auto-Regressive Integrated Moving Average Fault Detection (ARIMAFD) [44] model has been proposed as a variant of ARIMA for the streaming data. It is a computationally efficient model for online anomaly detection. However, it still follows the same assumption as ARIMA and assumes the input time series is weakly stationary. Similar to the online methods in the proximity-based category, its update strategy does not account for concept drift. The benefits and drawbacks of the predictive-based algorithms are summarized in Table 3.

Table 3 Predictive-based anomaly detection methods

Algorithm	**Advantages**	**Disadvantages**
Classification-based		
OCSVM [70]	Easy to implement	High time complexity
		Unstable performance due to the imbalanced dataset
THOC [71]	Competitive performance	High time complexity
	Using temporal information	Requires a large training set
iForest [53]	Efficient in both training and testing phases	Not effective when the dataset contains too many anomalies
	High accuracy	
Extended iForest [35]	Further improved the accuracy	Not effective when the dataset contains too many anomalies
Carletti et.al [16]	Provided interpretable feature selection methods	Not effective when the dataset contains too many anomalies
	Further improved the accuracy	
Regression-based		

Algorithm	**Advantages**	**Disadvantages**
ARMA [22]	Simple and easy to implement	Requires stationary time series to be stationary
	Low time efficiency	
ARIMA [9]	Low time efficiency	Requires time series to be weakly stationary
	Supports weakly stationary time series	
ARIMAFD [44]	Low time efficiency	Requires time series to be weakly stationary
	Online updates for steaming data	Concept drift is not considered in the update process

Although many classification algorithms exist, they often find it challenging to attain generally accurate results due to imbalanced data. Isolation-based ensemble methods can achieve high accuracy but are only effective when the dataset contains fewer anomalies. However, ensemble methods have the potential to be combined with concept drift detection methods. It allows us to update the base detectors based on the drift detection as shown in Fig. 1. Combining ensemble-based anomaly detection and concept drift detection methods can be a direction to investigate in the future.

Regarding the regression-based methods, a common shortcoming is that anomalies in the training set could impact the reliability of the regression methods. Also, most of them rely on the assumption of stationary data distribution. In addition, regression methods identify anomalies by predicting the data's trend. If concept drifts cause a change in the long-term trend of data, the efficacy of these algorithms is likely to degrade. Therefore, making them adapt to the concept drift is quite challenging.

2.2 Statistical Methods

Most statistical methods are probability-based techniques. The fundamental assumption of statistical methods is that anomalies often occur in regions with low probability when constructing a stochastic model from historical data instances. Each testing data will be compared to the model. If the data is statistically compatible with the model, it will be labeled as normal. Otherwise, the data is labeled as an anomaly. Parametric and non-parametric are two sub-categories of techniques of statistical models. The difference resides in the assumptions about data distribution. In parametric statistics assumptions are made about the distribution of data instances. Specifically, data instances should always be generated from a particular distribution. In contrast, non-parametric statistics are not founded on any premises of the instance distribution.

2.2.1 Parametric Methods

As a standard parametric technique, Gaussian model-based methods (G) presume that instances are generated from the Gaussian distribution. Mean value and standard division are two important parameters that control the probability density functions of Gaussian distribution. For each instance, $\mathbf{q}$, Gaussian model-based methods use the distance of it to the mean value to calculate the anomaly score. A mixture of parametric distribution-based techniques assumes that normal instances are drawn from a mixture of parametric statistical distributions. The Gaussian Mixture Model (GMM) method is a variant of traditional Gaussian model-based methods. It assumes that typical instances are generated by a mixture of parametric Gaussian distribution models. The overall probability of each instance $\mathbf{q}$ is calculated by the sum of probability density of $\mathbf{q}$ in each Gaussian distribution. If the overall probability is too small, the test instance does not belong to any parametric Gaussian distributions. Therefore, $\mathbf{q}$ will be declared as an anomaly. As a variant of GMM, the Deep Autoencoding Gaussian Mixture Model (DAGMM) [83] is intended for large, high-dimensional datasets. It employs the autoencoder method and seeks to preserve critical information in the low-dimensional space while performing dimensionality reduction. Consequently, GMM can handle high-dimensional data. Nonetheless, its adaptability to concept drift has not been considered. Autoencoder is a deep learning technique which will be discussed in the next subsection.

2.2.2 Non-parametric Methods

Regarding non-parametric statistical techniques, histogram-based methods are the most commonly used for identifying the anomalies based on the probability density. The traditional histogram method can only work in univariate datasets. This method begins by creating a histogram. For each test instance, the probability density of their corresponding bin is used to identify them. However, utilizing typical histogram methods to identify multivariate anomalies is challenging since they cannot account for the interactions between multiple variables.

Histogram-Based Outlier Score (HBOS) [31] has the flexibility to be applied in multivariate problems. To ensure each dimension has the same contribution to the final result, we must normalize each feature to the range [0,1]. The HBOS method builds a univariate histogram for every single feature and combines their probability density to produce the final score.

As another histogram-based method, Lightweight Online Detector of Anomalies (LODA) [63] has been proposed as an online ensemble anomaly detection method. This method is also considered as a clustering-based method. The probability density is approximated by projecting the data points onto a single vector and constructing an equal-width histogram. This method points out that it online adjusts the optimal number of histogram bins by optimizing the penalized maximum likelihood method [11]. Consequently, this method has some potential to address concept drift, although this has not been demonstrated through experiments yet. Experiment results from [58] indicate that LODA is one of the most sensitive algorithms to concept drift. Therefore, its adaptability with concept drift must be further investigated.

Furthermore, kernel-based methods are a subset of non-parametric methods. To detect anomalies, these methods use kernel functions to estimate probability density. These algorithms typically have a very high computational cost and a lengthy processing time. Therefore they may not be suitable for edge computing. The benefits and drawbacks of statistical methods are summarized in Table 4.

Table 4 Statistical anomaly detection methods

Algorithm	Advantages	Disadvantages
Parametric methods		
GMM-based [66, 83]	Easy to understand and implement	Rely on the assumption of data distribution
	Easy to interpretable	
Non-parametric methods		
Histogram-based [31, 63]	Easy to understand and implement	Sensitive to the bin width
	Easy to interpretable	The structure is too simple to handle large or high-dimensional dataset

Algorithm	Advantages	Disadvantages
Kernel-based [52, 74]	Easy to understand and implement	Require high computational cost
	Easy to interpretable	Long processing time

Although statistical algorithms are simple and easily interpretable, their shortcomings are also readily apparent. In parametric methods, the distribution of the data is always assumed. When concept drift alters data distribution, the parametric algorithm will become inaccurate. Non-parametric methods are either sensitive to the parameters or require high computational costs and long processing times. Consequently, there have been less studies in recent years.

2.3 Deep Learning Methods

These techniques have recently shown extraordinary proficiency in identifying expressive representations for high-dimensional data. Deep learning for anomaly detection entails training neural networks to develop feature representations or directly compute anomaly scores. There are a growing number of studies that give an exhaustive overview of deep anomaly detection techniques [18, 46, 60, 75]. A portion of the research indicates that deep anomaly detection methods outperform standard anomaly detection in various real-world applications involving difficult detection problems. Typical deep learning architecture contains recurrent neural network (RNN) [78], autoencoder (AE), and generative models [33]. Among them, RNN and its variants Long Short-Term Memory (LSTM) [37] and Gated Recurrent Unit (GRU) [23] show their benefit in processing time series and other sequence data. Each of these architectures can be used to learn and predict data. They can be used as the predictive-based methods and detect anomalies by comparing the prediction results and actual values, similar to a regression model.

In addition, several generic normality feature learning models detect anomalies by applying the above architectures. The models learn the representations of data instances. During the learning process, they capture the underlying data regularities of normal instances. These underlying regularities can be used to detect anomalies. These algorithms can be categorized as AE-based, Generative Adversarial Network (GAN)-based, and self-supervised learning-based.

Autoencoder-based AE-based methods aim to discover a low-dimensional feature representation for data instances and ensure that the instances can be reconstructed accurately. Generally, the normal instances should be the primary training samples. During the training, the encoder should keep the information relevant to normal instances to minimize reconstruction errors. Therefore, the assumption of AE-based methods is that anomalies are more difficult to reconstruct from compressed space than normal instances [60]. Commonly, the Mean Squared Error loss function is used to measure the reconstruction error and compute the anomaly score. Common techniques include LSTM-AE [57, 60] and GRU-AE [26] that combine AE with sequence deep learning architectures. These techniques enable the detection of anomalies in univariate and multivariate time series.

Another type of AE-based approach has two main hidden layers. AE will learn the low-dimensional representation of high-dimensional data input in the first hidden layer. After that, the low-dimensional representation will be considered as the label. The model will learn to predict these learned representations in the second hidden layer. The method denoising AE [55] applies RNN in each hidden layer. The first RNN is used to understand normal patterns of high-dimensional data, and the second RNN learns the prediction. This approach can also be considered as a self-supervised-based model.

GAN-based These models are intended to comprehend the actual distribution of data, thereby generating new instances. A generator and a discriminator are two essential components of GAN. The generator aims to find the learned latent feature space for instances. Then it produces fake instances, making them resemble the original data as closely as possible. The purpose of the discriminator is to differentiate between original and fake instances. These two components operate against each other, thereby enhancing their learning capabilities. Since the latent space analysis utilizes the training data’s underlying distribution, anomalies cannot have comparable accuracy to normal instances. Therefore, GAN-based approaches are founded on the fundamental premise that normal instances can be more easily generated from the generator than the anomalies. Residual and discrimination lossses are two values that measure the anomaly score. The discrimination loss is determined by matching features in the discriminator, and the residual loss is the disparity between the actual and fake instances.

Self-supervised These models learn the feature representations of previous instances. After that, they utilize the learned representations as content to predict current data instances. Video frame prediction is a typical application. Predictability models can use the previous video sequence to predict the upcoming frame [38]. Similarly, self-supervised models should be applicable in time series to predict the data at the subsequent time stamp. Commonly, predictability models assume that normal data are more temporally predictable than anomalies. Similar to regression methods, the anomaly score is calculated by comparing the predicted output and an actual new instance.

In addition to the aforementioned models, deep learning method can also be combined with statistical and machine learning anomaly detection algorithms. Its primary function is to achieve efficient feature learning and hierarchical feature extraction. Thus, valuable features are provided for the two categories of algorithms. During the process of extracting features, deep learning can also reduce the data dimension, enabling machine learning or statistical techniques to better manage high-dimensional data. This category contains methods that we have previously examined, such as DAGMM and Deep SVDD.

As summarized in Table 5, deep learning models have some general drawbacks. The main advantage of deep learning algorithms is the ability to handle high-dimensional data. However, it requires large computing power, which may be challenging for a single edge node. At the same time, deep learning models generally demand large training datasets. In contrast to cloud computing, a single edge node only collects data from nearby devices. Consequently, the quantity of data is relatively small. It causes a considerable amount of time to collect the data before training the model. Meanwhile, the training phase in the deep learning model also requires considerable time. As shown in Fig. 1, the ideal anomaly detection model should be updated in real time based on concept drift detection results to ensure it has long-term accurate performance. A longer training time might lead to a longer model update time, resulting in a less effective model. In this scenario, the model may not produce a good and timely indication for data quality. Moreover, its lack of interpretability is not conducive to the analysis of anomalies at a later stage.

Table 5 Deep learning anomaly detection methods

Algorithm	Advantage	Disadvantage
AE-based [28, 42, 65, 81]	High accuracy	High computational cost
GAN-based [7, 19, 48–50]	Can handle high-dimensional data effectively	High time complexity
Self-supervised [7, 40, 48– 50, 82]	Application agnostic	Require a large and high-dimensional dataset
		Poor interpretability

3 Concept Drift Detection

As illustrated in Fig. 1, anomaly detection algorithm should be updated based on the concept drift detection result in the proposed framework. This is due to the fact that concept drift frequently occurs in time series as time passes, which will impact the performance of trained models. Detecting concept drift in time series data is essential for ensuring models' long-term stability. In this section, we analyze and compare two categories of widely utilized concept drift detection methods.

Concept drift refers to variations in data distribution that occur unexpectedly over time [54], which is characterized by [68, 76]. The concept drift emphasizes the change of the input data distribution that leads to a different functional relationship between the input and output data. Therefore, the existence of concept drift usually impacts the performance of the model [2, 8, 34]. Existing research indicates that concept drift detection methods can primarily be categorized into two groups: data distribution- and performance-based. They are applicable to the majority of existing learning models and have low computing complexity.

3.1 Data Distribution-Based Methods

In these methods, the time series will be divided into multiple windows according to the time. The data distribution similarity across distinct windows will then be estimated using distance measures. Existing literature provides a variety of distance metrics applicable to univariate time series. The Hellinger distance [24], Kullback–Leibler divergence [41], and total variation distance [15] are the most frequently used algorithms. Regarding multivariate time series, the most commonly used algorithms include PCA and Mahalanobis distance. Data distribution-based algorithms have the benefit that they are simple to implement and comprehend. In addition, it can work with unlabeled data since this type of algorithm merely compares the similarity of the data distributions. Consequently, they are the most straightforward for anomaly detection problems where ground truths are typically lacking. Conversely, the disadvantages of this type of algorithm are also readily apparent. The purpose of concept drift detection is to mitigate the effect of data changes on the efficacy of the model. However, previous study [8] mentioned that certain changes in data distributions may have no effect on the efficacy of the model, which could result in false positives using this type of methods.

3.2 Performance-Based Methods

These methods aim to detect concept drifts when there is a substantial decline in model performance. These methods can be further categorized into the Statistical Process Control (SPC), window-based, and ensemble-based categories. SPC models monitor the online error rate to measure model output quality. The drift detection method (DDM) and the early drift detection method (EDDM) [6] are two prevalent performance-based methods. They are proficient at detecting model-degrading drifts. Similar to data distribution-based methods, window-based detectors divide the time series into multiple windows. They monitor the performance of the latest window's performance and compare it to previous windows' performance. AdaptiveWindowing (ADWIN) [10] employs adaptive sliding windows. It keeps monitoring the difference between two consecutive windows to detect the drift. This procedure is simple to implement and effective. As the third category, ensemble-based methods monitor the performance of each base detector to identify the overall model performance.

As shown in Table 6, although these algorithms are more focused on detecting concept drift that affects model performance, most of them require labeled datasets to train and evaluate the model's performance over time. Since datasets are typically unlabeled in anomaly detection problems, using these algorithms directly in anomaly detection methods is challenging. An unsupervised concept drift detection method [17] is developed by estimating pseudo-error. However, such approaches are still uncommon and have substantial research potential. For further studies of concept drift detection techniques, we refer the readers to a few surveys [3, 8, 29].

Table 6 Concept drift detection categories

Algorithm	Advantages	Disadvantages
Distribution-based [15, 24, 29, 41]	Simple and intuitive	High probability of false alarm
	Can be used for unlabeled data	
	High efficiency	
Performance-based [8, 17, 80]	High accuracy	Only a few algorithms can be used on unlabeled data

4 Discussions and Future Research Directions

As we mentioned in the introduction, our proposed anomaly detection framework should adapt to concept drifts to ensure long-term performance reliability in the context of edge computing, as illustrated in Fig. 1. In edge computing applications, the ideal model should use a small amount of training data to begin checking data quality as soon as feasible. We also need it to be able to process data efficiently and provide data quality reports for edge computing in a timely fashion.

This section analyzes and summarizes the efficacy of each category of anomaly detection. We will discuss the potential feasibility of them adaptive to concept drift and provide some future directions. The performance comparison is shown in Table 7 with qualitative ratings of poor (–), average (o), and excellent (+). We focus on the following challenges: **accuracy of detection**, which also takes into account various anomaly types; **stability**, which represents model performance stability across different databases; **training data**, which represents the training set size required by the model; **sensitivity**, which represents the model's sensitivity to parameters; and **speed**, which represents the model's computation time. The accuracy comparison is based on our analysis in Sect. 2 and the experimental results of related studies [13, 32, 62].

Table 7 Summary and comparison of anomaly detection methods

Category/attr.	**Accuracy**		**Stability**	**Training data**	**Sensitivity**	**Speed**
	Point/collective	**Contextual**				
Statistical						
Non-parametric	+	—	o	+	o	+
Parametric	+	—	—	+	+	+
Machine learning						
Proximity-based	+	o/—	o	o	o	o
Clustering-based	+	+	o	o	—	o
Predictive-based	+/o	—	o	o	+	+/o
Deep learning						
Predictive-based	o	+	o	—	+	—
AE-based	o	o	+	—	+	—

Statistical models are relatively simple to implement and have competitive accuracy performance on point and collective anomalies. In addition, the computation speed of statistical models is superior than other algorithms. However, the primary drawback of parametric statistical methods is that they presume the data follow a particular distribution. This assumption directly opposes the issue of concept drift. Regarding the non-parametric method, kernel-based methods have a very high computational cost and a lengthy processing time. Histogram-based method is typically applied only to stationary time series based on the simplicity of the structure. Also, it is sensitive to the width of the histogram bins. Without ground truth, determining the optimal bin width value is also challenging.

Machine learning algorithms are outperformed by deep learning techniques when detecting contextual anomalies. However, deep learning techniques perform poorly in detecting points and collective anomalies. As point and collective anomalies are the two most general and fundamental categories of anomaly, the research of adaptive anomaly detection methods mostly focuses on them. In addition, deep learning demands a large number of training instances, whereas anomaly detection in edge computing often begins with fewer training instances. Furthermore, lengthy training and testing duration need to be improved for deep learning algorithms. If deep learning algorithms are to be utilized in the future, the research into lightweight deep learning must be prioritized.

Regarding the category of machine learning, clustering-based methods obtain high accuracy in detecting all types of anomalies. However, clustering is tricky for time series anomaly detection problems. This is because we will need more data initially to form correct clusters. Therefore, defining the cluster number in real-world applications is difficult. Also, clustering requires a large amount of data to ensure a complete data distribution representation. For proximity-based techniques, it necessitates that all normal instances converge to a certain region, ensuring that they are at comparable distances from their *k*th nearest neighbor. Consequently, once concept drift modifies data distribution, the accuracy of such algorithms would be highly susceptible. Due to imbalanced data, it is difficult for predictive-based methods to produce generally accurate results. Several isolation-based methods can attain high precision and efficiency. However, they are only effective when fewer anomalies exist in the dataset.

Based on our analysis, we identified that the current anomaly detection techniques are susceptible to concept drift. Improving their adaptability and long-term stability is thus the primary challenge for these algorithms in edge computing applications. Therefore, future research need to focus on the feasibility of integrating them with the concept drift detection technique and how to construct the update strategy for the anomaly detection model so that it can learn from new data patterns. Meanwhile, future research should also concentrate on the following aspects: small training sets, fast training, and lightweight deep learning models, thus making anomaly detection methods to be more readily integrated with concept drift detection techniques. We also found that ensemble-based models have great potentials to expand their adaptability. Since the ensemble models consist of multiple base detectors, it is feasible to enhance its overall performance by updating some base detectors.

---

5 Conclusion

Based on the critical impact of data quality in the field of IoT and edge computing, this chapter presents existing unsupervised anomaly detection methods, the potential impacts of concept drifts and associated detection methods, and the necessity of extending the model’s adaptability to assure the model’s long-term stability. At the same time, the vision of an integrated framework for adaptive anomaly detection with concept drift detection is proposed to enhance their adaptability to new data, thus improving their long-term credibility. By comparing the anomaly detection methods, we analyzed the potential of various algorithms to extend their adaptivity with the concept drift. For the readers to gain a deeper comprehension of concept drift, we also provide an overview analysis and comparison of concept drift detection methods. Finally, the challenges of integrating them and the future research direction are summarized.

Acknowledgements

This project was partially supported by Te Hiranga Rū QuakeCoRE, an Aotearoa New Zealand Tertiary Education Commission-funded Centre. This is QuakeCoRE publication number 920.

References

1. Aggarwal, C.C.: Proximity-Based Outlier Detection, pp. 111–147. Springer International Publishing, Cham (2017)
2. Agrahari, S., Singh, A.K.: Concept drift detection in data stream mining: a literature review. J. King Saud Univ.-Comput. Inf. Sci. **34**(10), 9523–9540 (2022)
3. Alimohammadi, H., Chen, S.N.: Performance evaluation of outlier detection techniques in production timeseries: a systematic review and meta-analysis. Expert Syst. Appl. **191**, 116371 (2022)
4. Angiulli, F., Pizzuti, C.: Fast outlier detection in high dimensional spaces. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) Principles of Data Mining and Knowledge Discovery, 6th European Conference, PKDD 2002, Helsinki, 19–23 Aug 2002, Proceedings. Lecture Notes in Computer Science, vol. 2431, pp. 15–26. Springer (2002)
5. Angiulli, F., Pizzuti, C.: Outlier mining in large high-dimensional data sets. IEEE Trans. Knowl. Data Eng. **17**(2), 203–215 (2005)
6. Baena-Garcıa, M., del Campo-Ávila, J., Fidalgo, R., Bifet, A., Gavalda, R., Morales-Bueno, R.: Early drift detection method. In: Fourth International Workshop on Knowledge Discovery from Data Streams, vol. 6, pp. 77–86 (2006)
7. Bashar, M.A., Nayak, R.: Tanogan: time series anomaly detection with generative adversarial networks. In: 2020 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1778–1785. IEEE (2020)
8. Bayram, F., Ahmed, B.S., Kassler, A.: From concept drift to model degradation: an overview on performance-aware drift detectors. Knowl.-Based Syst. **245**, 108632 (2022)

9. Bianco, A., Garcia Ben, M., Martinez, E.J., Yohai, V.: Outlier detection in regression models with ARIMA errors using robust estimates. J. Forecast. **20**, 565–579 (2001)

10. Bifet, A., Gavalda, R.: Learning from time-changing data with adaptive windowing. In: Proceedings of the 2007 SIAM International Conference on Data Mining, pp. 443–448. SIAM (2007)

11. Birge, L., Rozenholc, Y.: How many bins should be put in a regular histogram. ESAIM: Probab. Stat. **10**, 24–45 (2006)
[MathSciNet]

12. Blázquez-García, A., Conde, A., Mori, U., Lozano, J.A.: A review on outlier/anomaly detection in time series data. ACM Comput. Surv. (CSUR) **54**(3), 1–33 (2021)

13. Braei, M., Wagner, S.: Anomaly detection in univariate time-series: a survey on the state-of-the-art (2020)

14. Breunig, M.M., Kriegel, H., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. In: Chen, W., Naughton, J.F., Bernstein, P.A. (eds.) Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, 16–18 May 2000, Dallas, pp. 93–104. ACM (2000)

15. Brown, M.: The distribution of total variation distance, with applications to simultaneous confidence intervals. Comput. Oper. Res. **22**(4), 373–381 (1995)
[MathSciNet]

16. Carletti, M., Terzi, M., Susto, G.A.: Interpretable anomaly detection with DIFFI: depth-based feature importance of isolation forest. Eng. Appl. Artif. Intell. **119**, 105730 (2023)

17. Cerquitelli, T., Proto, S., Ventura, F., Apiletti, D., Baralis, E.: Towards a real-time unsupervised estimation of predictive model degradation. In: Proceedings of Real-Time Business Intelligence and Analytics, pp. 1–6 (2019)

18. Chalapathy, R., Chawla, S.: Deep learning for anomaly detection: a survey. arXiv preprint arXiv:1901.03407 (2019)

19. Challu, C.I., Jiang, P., Wu, Y.N., Callot, L.: Deep generative model with hierarchical latent factors for time series anomaly detection. In: International Conference on Artificial Intelligence and Statistics, pp. 1643–1654. PMLR (2022)

20. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. **41**(3), 15:1–15:58 (2009)

21. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection for discrete sequences: a survey. IEEE Trans. Knowl. Data Eng. **24**(5), 823–839 (2012)

22. Chang, I., Tiao, G.C., Chen, C.: Estimation of time series parameters in the presence of outliers. Technometrics **30**(2), 193–204 (1988)
[MathSciNet]

23. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 (2014)
24. Cieslak, D.A., Hoens, T.R., Chawla, N.V., Kegelmeyer, W.P.: Hellinger distance decision trees are robust and skew-insensitive. Data Min. Knowl. Discov. **24**(1), 136–158 (2012) [MathSciNet]
25. Cook, A.A., Misirli, G., Fan, Z.: Anomaly detection for IoT time-series data: a survey. IEEE Internet Things J. **7**, 6481–6494 (2020)
26. Cowton, J., Kyriazakis, I., Plötz, T., Bacardit, J.: A combined deep learning GRU-autoencoder for the early detection of respiratory disease in pigs using multiple environmental sensors. Sensors **18**(8), 2521 (2018)
27. Du, H., Zhao, S., Zhang, D., Wu, J.: Novel clustering-based approach for local outlier detection. In: 2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 802–811. IEEE (2016)
28. Gao, H., Qiu, B., Barroso, R.J.D., Hussain, W., Xu, Y., Wang, X.: Tsmae: a novel anomaly detection approach for internet of things time series data using memory-augmented autoencoder. IEEE Trans. Netw. Sci. Eng. **10**, 2978–2990 (2022) [MathSciNet]
29. Goldenberg, I., Webb, G.I.: Survey of distance measures for quantifying concept drift and shift in numeric data. Knowl. Inf. Syst. **60**(2), 591–615 (2019)
30. Goldstein, M.: Fastlof: an expectation-maximization based local outlier detection algorithm. In: Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012), pp. 2282–2285 (2012)
31. Goldstein, M., Dengel, A.: Histogram-based outlier score (HBOS): a fast unsupervised anomaly detection algorithm. KI-2012: Poster Demo Track **1**, 59–63 (2012)
32. Goldstein, M., Uchida, S.: A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. PloS One **11**(4), e0152173 (2016)
33. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Adv. Neural Inf. Process. Syst. **27**, 2672–2680 (2014)
34. Guo, H., Li, H., Ren, Q., Wang, W.: Concept drift type identification based on multi-sliding windows. Inf. Sci. **585**, 1–23 (2022)
35. Hariri, S., Kind, M.C., Brunner, R.J.: Extended isolation forest. IEEE Trans. Knowl. Data Eng. **33**(4), 1479–1489 (2019)
36. He, Z., Xu, X., Deng, S.: Discovering cluster-based local outliers. Pattern Recogn. Lett. **24**(9–10), 1641–1650 (2003)
37. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)

38. Hsieh, J.-T., Liu, B., Huang, D.-A., Fei-Fei, L.F., Niebles, J.C.: Learning to decompose and disentangle representations for video prediction. Adv. Neural Inf. Process. Syst. **31**, 515–524 (2018)

39. Huo, W., Wang, W., Li, W.: Anomalydetect: an online distance-based anomaly detection algorithm. In: Web Services–ICWS 2019: 26th International Conference, Held as Part of the Services Conference Federation, SCF 2019, San Diego, 25–30 June 2019, Proceedings, vol. 26, pp. 63–79. Springer (2019)

40. Jiao, Y., Yang, K., Song, D., Tao, D.: Timeautoad: autonomous anomaly detection with self-supervised contrastive loss for multivariate time series. IEEE Trans. Netw. Sci. Eng. **9**(3), 1604–1619 (2022)

41. Joyce, J.M.: Kullback-leibler divergence. In: Lovric, M. (ed.) International Encyclopedia of Statistical Science, pp. 720–722. Springer, Berlin/Heidelberg (2011)

42. Kieu, T., Yang, B., Guo, C., Jensen, C.S.: Outlier detection for time series with recurrent autoencoder ensembles. In: IJCAI, pp. 2725–2732 (2019)

43. Koupaie, H.M., Ibrahim, S., Hosseinkhani, J.: Outlier detection in stream data by clustering method. Int. J. Adv. Comput. Sci. Inf. Technol. **2**(3), 25–34 (2013)

44. Kozitsin, V., Katser, I., Lakontsev, D.: Online forecasting and anomaly detection based on the ARIMA model. Appl. Sci. **11**(7), 3194 (2021)

45. Kriegel, H.-P., Schubert, M., Zimek, A.: Angle-based outlier detection in high-dimensional data. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 444–452 (2008)

46. Kwon, D., Kim, H., Kim, J., Suh, S.C., Kim, I., Kim, K.J.: A survey of deep learning-based network anomaly detection. Clust. Comput. **22**(1), 949–961 (2019)

47. Langfu, C., Zhang, Q., Yan, S., Liman, Y., Yixuan, W., Junle, W., Chenggang, B.: A method for satellite time series anomaly detection based on Fast-DTW and improved-KNN. Chin. J. Aeronaut. **36**(2), 149–159 (2023)

48. Lee, C.-K., Cheon, Y.-J., Hwang, W.-Y.: Studies on the GAN-based anomaly detection methods for the time series data. IEEE Access **9**, 73201–73215 (2021)

49. Li, D., Chen, D., Goh, J., Ng, S.-K.: Anomaly detection with generative adversarial networks for multivariate time series. arXiv preprint arXiv:1809.04758 (2018)

50. Li, D., Chen, D., Jin, B., Shi, L., Goh, J., Ng, S.-K.: Mad-GAN: multivariate anomaly detection for time series data with generative adversarial networks. In: Artificial Neural Networks and Machine Learning–ICANN 2019: Text and Time Series: 28th International Conference on Artificial Neural Networks, Munich, 17–19 Sep 2019, Proceedings, Part IV, pp. 703–716. Springer (2019)

51. Li, J., Izakian, H., Pedrycz, W., Jamal, I.: Clustering-based anomaly detection in multivariate time series data. Appl. Soft Comput. **100**, 106919 (2021)

52. Liu, F., Yu, Y., Song, P., Fan, Y., Tong, X.: Scalable KDE-based top-n local outlier detection over large-scale data streams. Knowl.-Based Syst. **204**, 106186 (2020)

53. Liu, F.T., Ting, K.M., Zhou, Z.: Isolation-based anomaly detection. ACM Trans. Knowl. Discov. Data **6**(1), 3:1–3:39 (2012)

54. Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., Zhang, G.: Learning under concept drift: a review. CoRR abs/2004.05785 (2020)

55. Lu, W., Cheng, Y., Xiao, C., Chang, S., Huang, S., Liang, B., Huang, T.: Unsupervised sequential outlier detection with deep architectures. IEEE Trans. Image Process. **26**(9), 4321–4330 (2017) [MathSciNet]

56. Ma, R., Pang, G., Chen, L., van den Hengel, A.: Deep graph-level anomaly detection by glocal knowledge distillation. In: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, pp. 704–714 (2022)

57. Malhotra, P., Vig, L., Shroff, G., Agarwal, P., et al.: Long short term memory networks for anomaly detection in time series. In: Proceedings, vol. 89, pp. 89–94 (2015)

58. Mendelson, S., Lerner, B.: Online cluster drift detection for novelty detection in data streams. In: 2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 171–178. IEEE (2020)

59. Na, G.S., Kim, D., Yu, H.: Dilof: effective and memory efficient local outlier detection in data streams. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1993–2002 (2018)

60. Pang, G., Shen, C., Cao, L., Hengel, A.V.D.: Deep learning for anomaly detection: a review. ACM Comput. Surv. (CSUR) **54**(2), 1–38 (2021)

61. Pang, G., Ting, K.M., Albrecht, D.: Lesinn: detecting anomalies by identifying least similar nearest neighbours. In: 2015 IEEE International Conference on Data Mining Workshop (ICDMW), pp. 623–630. IEEE (2015)

62. Paparrizos, J., Kang, Y., Boniol, P., Tsay, R.S., Palpanas, T., Franklin, M.J.: TSB-UAD: an end-to-end benchmark suite for univariate time-series anomaly detection. Proc. VLDB Endow. **15**(8), 1697–1711 (2022)

63. Pevný, T.: Loda: lightweight on-line detector of anomalies. Mach. Learn. **102**(2), 275–304 (2016) [MathSciNet]

64. Pokrajac, D., Lazarevic, A., Latecki, L.J.: Incremental local outlier detection for data streams. In: 2007 IEEE Symposium on Computational Intelligence and Data Mining, pp. 504–515. IEEE (2007)

65. Provotar, O.I., Linder, Y.M., Veres, M.M.: Unsupervised anomaly detection in time series using LSTM-based autoencoders. In: 2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT), pp. 513–517. IEEE (2019)

66. Rasmussen, C.: The infinite Gaussian mixture model. In: Solla, S., Leen, T., Müller, K., (eds.) Advances in Neural Information Processing Systems, vol. 12. MIT Press, USA (1999)

67. Ruff, L., Görnitz, N., Deecke, L., Siddiqui, S.A., Vandermeulen, R.A., Binder, A., Müller, E., Kloft, M.: Deep one-class classification. In: Dy, J.G., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, 10–15 July 2018. Proceedings of Machine Learning Research, vol. 80, pp. 4390–4399. PMLR (2018)

68. Sato, D.M.V., De Freitas, S.C., Barddal, J.P., Scalabrin, E.E.: A survey on concept drift in process mining. ACM Comput. Surv. (CSUR) **54**(9), 1–38 (2021)

69. Saurav, S., Malhotra, P., TV, V., Gugulothu, N., Vig, L., Agarwal, P., Shroff, G.: Online anomaly detection with concept drift adaptation using recurrent neural networks. In: Proceedings of the ACM India Joint International Conference on Data Science and Management of Data, pp. 78–87 (2018)

70. Schölkopf, B., Smola, A.J., Williamson, R.C., Bartlett, P.L.: New support vector algorithms. Neural Comput. **12**(5), 1207–1245 (2000)

71. Shen, L., Li, Z., Kwok, J.: Timeseries anomaly detection using temporal hierarchical one-class network. Adv. Neural Inf. Process. Syst. **33**, 13016–13026 (2020)

72. Silverman, B.W., Jones, M.C.: E. fix and j.l. hodges (1951): An important contribution to nonparametric discriminant analysis and density estimation: commentary on fix and hodges (1951). Int. Stat. Rev./Revue Internationale de Statistique **2336**, 233–238 (1989)

73. Tang, J., Chen, Z., Fu, A.W.-C., Cheung, D.W.: Enhancing effectiveness of outlier detections for low density patterns. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 535–548. Springer (2002)

74. Ting, K.M., Xu, B.-C., Washio, T., Zhou, Z.-H.: Isolation distributional kernel: a new tool for kernel based anomaly detection. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 198–206 (2020)

75. Wang, R., Nie, K., Wang, T., Yang, Y., Long, B.: Deep learning for anomaly detection. In: Proceedings of the 13th International Conference on Web Search and Data Mining, pp. 894–896 (2020)

76. Webb, G.I., Hyde, R., Cao, H., Nguyen, H.L., Petitjean, F.: Characterizing concept drift. Data Mining Knowl. Discov. **30**(4), 964–994 (2016)
[MathSciNet]

77. Wells, J.R., Ting, K.M., Washio, T.: Linearn: a new approach to nearest neighbour density estimator. Pattern Recogn. **47**(8), 2702–2720 (2014)

78. Williams, R.J., Zipser, D.: A learning algorithm for continually running fully recurrent neural networks. Neural Comput. **1**(2), 270–280 (1989)

79. Yang, K., Kpotufe, S., Feamster, N.: An efficient one-class SVM for anomaly detection in the internet of things. CoRR abs/2104.11146 (2021)

80. Yang, L., Shami, A.: A lightweight concept drift detection and adaptation framework for iot data streams. IEEE Internet Things Mag. **4**(2), 96–101 (2021)

81. Yin, C., Zhang, S., Wang, J., Xiong, N.N.: Anomaly detection based on convolutional recurrent autoencoder for IoT time series. IEEE Trans. Syst. Man Cybern.: Syst. **52**(1), 112–122 (2020)

82. Zhang, Y., Wang, J., Chen, Y., Yu, H., Qin, T.: Adaptive memory networks with self-supervised learning for unsupervised anomaly detection. IEEE Trans. Knowl. Data Eng. **35**, 12068–12080 (2022)

83. Zong, B., Song, Q., Min, M.R., Cheng, W., Lumezanu, C., Cho, D., Chen, H.: Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, 30 Apr–3 May 2018, Conference Track Proceedings. OpenReview.net (2018)

OceanofPDF.com

Part II
Security and Privacy Paradigm

OceanofPDF.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
S. Pal et al. (eds.), *IoT Edge Intelligence*, Internet of Things: Technology, Communications and Computing
https://doi.org/10.1007/978-3-031-58388-9_7

Secure, Trusted, Privacy-Protected Data Exchange in an Edge-Cloud Continuum Environment

Salvador Cuñat[1] ✉, Matilde Julian[1], Andreu Belsa[1], Clara I. Valero[1], Manuel Esteve[1] and Carlos E. Palau[1]
(1) Universitat Politècnica de Valencia, Valencia, Spain

✉ **Salvador Cuñat**
Email: salcuane@upv.es

Keywords Data Spaces – Data exchange – Edge-Cloud Continuum – Data Fabric – Data as a product

1 Introduction

The ability to enhance cost-efficiency and the reduction of the time needed to improve business processes and workflows provide the most value for data-driven industrial value networks. These networks have an increasing dependency on creating and maintaining stable data value ecosystems, complete with trusted sharing and effective distributed data processing over common Data Spaces. These do not come without shortcomings, however. Some of the most common difficulties associated with the creation of data ecosystems are the high variety of use cases, which implies the use of different processing and storage flows, the huge volumes of data that have to be processed, the maintenance of the value, the quality of the data, and the heterogeneity of the data sources [1].

In the contemporary interconnected world, the concept of Data Space has emerged as a crucial domain for technological advancements and innovative solutions. Considering the Computing Continuum, federation of systems, and data collection/trading, a distinct perspective arises regarding the immense potential of these data exchange ecosystems. To facilitate the expansion and scalability of Data Spaces, it becomes imperative to establish a continuum that transcends both spatial and temporal dimensions [2].

Spatially, the focus lies on enabling seamless data exchange across a diverse range of processors, encompassing IoT, Edge Computing (EC), and Cloud platforms. This dimension also involves the integration of smart solutions into various domains. By facilitating the flow of data across heterogeneous systems, this spatial perspective forms the foundation for transformative advancements.

Simultaneously, the temporal dimension plays a pivotal role in the evolution of Data Spaces. It addresses the dynamic nature of data and the entities it encompasses. The time dimension ensures the adaptability, expansion, and evolution of Data Spaces over time. This temporal perspective enables continuous growth, accommodates new data sources, and fosters the development of innovative applications [3].

The entire content of the chapter can be organized into seven different sections. Starting with an introduction where the contents of the chapter are presented, followed by a section explaining the different motivations and technologies that enabled the DaaP approach [4, 5]. Afterwards, the main body of the chapter starts. The Data Exchange section comes first, which is needed to explain and go over both the existing Data Spaces that will be used and the Data as a Service (DaaS) approach. Following Data Exchange comes the Modern Data Management Frameworks section, going over the state-of-the-art data platform concepts. To start, the Data Fabric concept is introduced, presenting a technical architecture that will be heavily related to the Data Mesh concept, since both approaches can have significant synergies. In parallel to this, the DaaP approach, which works alongside the Data Mesh, is discussed. The combination of these approaches is fulfilled, culminating in the use case presented in the chapter: the RE4DY project. RE4DY is a European initiative meant to enhance individual and value chain flexibility, employing the already mentioned synergy between the DaaP approach and the Data Fabric architecture. This section covers the technical approach of the project, as well as the different pilots that participate. The chapter ends with the conclusions and parting thoughts extracted.

---

2 Motivations and Related Work

The aim of this section is to provide an overview of the technologies needed to define the basis and the context that enable the DaaP technological approach. First, the Edge-Cloud Continuum encompasses the smooth integration of Cloud Computing (CC) resources and EC devices, enabling organizations to manage efficient data processing and analytic services [6]. It combines the advantages of both Cloud and EC, empowering organizations to leverage the scalability and computational power of the Cloud while minimizing delays and maximizing real-time data processing capabilities at the Edge. Second, the interoperability concept is presented, as it is key that the different services and data sources are able to share data with a consistent understanding of its meaning. Third, the Data Pipelines are responsible for collecting, transforming, and transporting data across the network. They ensure that data is ingested, processed, and analyzed efficiently, enabling organizations to obtain valuable insights.

2.1 Edge-Cloud Continuum

CC is currently one of the main elements of the technology landscape, allowing data and service sharing as well as a more efficient use of computing resources, which become easily scalable and replicable. The data is processed and stored within the Cloud infrastructure, minimizing the need for extensive integration with a high number of physical locations or heterogeneous endpoints. CC has dominated the arena of IT world during the past decades [7]. Despite having faced some challenges and fine-tuning (such as the CDNs, the client-server vs subscription or non-state approaches, REST vs SOAP in the web field, etc.), it has remained the reference in the vast majority of modern IT deployments. However, the advent of the new generation of the Internet, characterized by ever-increasing demands on bandwidth, latency, and computing power, has brought to light some flagrant shortcomings of this paradigm. The main limitations that current CC based systems are facing are the following:

- *Latency*: In CC, the processing of data often takes place at distant locations, spanning millions of kilometers away from the actual site where the action needs to be executed. As a result, there are significant delays of around 200 to 500 milliseconds in communication. These delays can be particularly problematic in critical applications such as factory automation, clinical surgery, and intelligent transport systems [8], where even minor delays can lead to disastrous outcomes.
- *Bandwidth*: The cost of transmitting video streams over the network while maintaining acceptable data rates has become prohibitively expensive for numerous applications [9], primarily due to the substantial bandwidth requirements they impose. However, it is important to note that the issue of limited bandwidth availability is not solely caused by multimedia applications. Even web pages [10] and sensor streams, due to their significant data content, have overwhelming transmission capacities, leading to constraints in data transmission capabilities.
- *Privacy*: Under the current CC model, data is required to travel to remote cloud servers located at considerable distances. This journey often involves traversing multiple networks and intermediate elements, raising concerns about data security along the way. A potential solution to address these challenges involves implementing a mechanism that allows for data reduction or filtering prior to reaching the Cloud. Moreover, enhancing the system with reasoning capabilities can serve as a network firewall, permitting only essential, secure, and minimized data to be transmitted, effectively maintaining control over sensitive information.
- *Dependability on Network Quality Connection and Cloud Services Provider Availability*: Under the current CC model, data is required to travel to remote cloud servers located at considerable distances. This journey often involves traversing multiple networks and intermediate elements, raising concerns about data security along the way. A potential solution to address these challenges involves implementing a mechanism that allows for data reduction or filtering prior to reaching the Cloud. Moreover, enhancing the system with reasoning capabilities can serve as a network firewall, permitting only essential, secure, and minimized data to be transmitted, effectively maintaining control over sensitive information.

All the previous issues were just at their birth. The same (and further) problems will be enlarged in the aftermath of the new wave of emerging technologies.

There is a significant debate surrounding the specific language used in the realm of EC. While attempting to establish a precise and official description of EC, various instances can be found in published works. Despite variations in technical details, primarily caused by the absence of a definitive authority or established standard, there is consensus on several crucial aspects: EC involves the relocation of computational capabilities closer to the origin of data. The Industrial Internet Consortium defines it as follows: "*EC is a decentralized computing infrastructure in which computing resources and application services can be distributed along the communication path from the data source to the Cloud*" [11]. For Mostafavi [12], "*EC enables services to exploit the proximity of devices by providing computational resources closer to end nodes, therefore enabling ultra-low latency and high data rate communication*". According to M. Satyanarayanan, an authoritative voice in the field, EC is "*a new paradigm in which substantial computing and storage resources—variously referred to as cloudlets, micro datacenters, or fog nodes are placed at the Internet's Edge in close proximity to mobile devices or sensors*" [13].

More recently, the development of EC has placed data processing and storage capabilities closer to the edge devices. This way, data can be processed closer to the sources, which results in increased data privacy and security, as well as better performance and less network usage and latency.

The combination of edge and cloud resources and services is known as the Digital Continuum or the Computing Continuum [14, 15]. This paradigm proposes a decentralized approach where the collection, storage, and processing of data are done in a seamless way from Edge to Cloud, and it reflects the integration of data processing services across all parts of the system. Cloud technologies play a crucial role in facilitating a seamless flow of data processing capabilities from Cloud to Edge, ensuring broad accessibility for stakeholders within the European Cloud ecosystem.

As can be seen in Fig. 1, in the Edge-to-Cloud Computing Continuum approach data is generated and prepared at the Edge, processed then by intermediate nodes and services and, if necessary, transferred to the Cloud. The structure of Edge-Cloud Continuum computing can be separated into three main levels:

- The *front-end level* encompasses various endpoint devices, including sensors, actuators, mobile devices (such as smartphones, tablets, and wearables), personal computers, and other data-generating elements like cameras and Bluetooth devices. In most deployments, these components are not equipped to handle extensive computing requirements, if any at all. Consequently, their primary role within this layer is to act as data forwarders, transferring data from the source to subsequent stages of the system for processing and analysis.

- The *near-end level* plays a crucial role in the context of EC. It serves as a significant aspect of the paradigm by executing specific tasks aimed at reducing traffic toward the Cloud and attaining the benefits mentioned earlier. Typical tasks assigned to this level include filtering, preprocessing, aggregation, content caching, device management, and privacy protection. On the other hand, the back-end level represents the traditional element centered around CC. This is where most of the bulk processing, storage, and analysis of data takes place.
- The *back-end level* pertains to the traditional aspect of CC, focusing on centralized processing and storage operations. This layer, which can be located in a single remote site or distributed across multiple locations, serves as the primary host for the main workload associated with cloud services. In certain cases, this level may also act as a centralized controller within the EC environment, overseeing, and coordinating various tasks and resources.

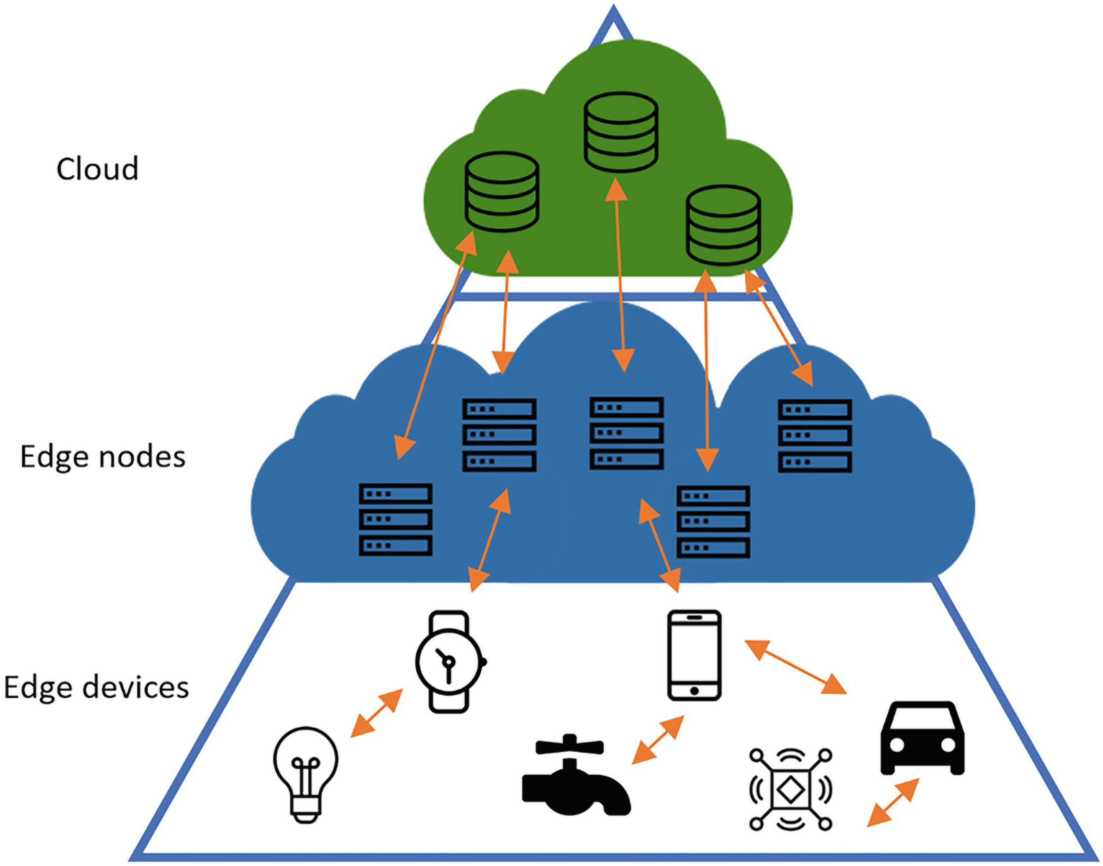


Fig. 1 Edge-to-Cloud Computing Continuum approach and its advantages

2.2 Interoperability and Data Processing

Interoperability among different computer systems is the ability to share services and data between them. Since each software solution provides its own infrastructure, devices, APIs, and data formats, achieving interoperability entails having effective communication and coordination on multiple levels between those components and systems that together compose a uniform platform at a larger scale. In order to be interoperable, two or more systems must be able to exchange, interpret, and present shared data in a way that is understood by all [16].

There are two types of data interoperability: syntactic interoperability and semantic interoperability. Syntactic interoperability is based on the adoption of a common data format and communication protocol and is a prerequisite to semantic interoperability. Semantic interoperability refers to the ability of computer systems to exchange data with a shared and unambiguous meaning [17]. In order to achieve interoperability and to allow integration with other services, it is very important to have a previously well-defined data representation standard, as well as approved ontologies.

With interoperability defined and achieved, the next step is data processing, which can be defined as the collection and manipulation of digital data to produce meaningful and useful information. Data processing involves a diverse array of activities conducted on data, utilizing both manual and automated methods. It encompasses various stages, such as data collection, organization, structuring, storage, dissemination, availability, etc. Data protection and security measures must be enforced when processing personal data wholly or partly by automated means, as well as in the case of non-automated processing, if it is part of a structured filing system.

2.3 Data Storage

The two types of overall data that can be found in organizations are operational data and analytical data. Operational data can be defined as the data that keeps the current state of the business. Sometimes referred to as "data on the inside," it represents the data that business applications use to serve the end users. This data is typically stored in systems that are optimized for data access, such as OnLine Transaction Processing (OLTP) [18].

Analytical data is the historical, aggregated view of operational data. This kind of data is usually stored in systems like Data Warehouses or Data Lakes. In this case, known as "data on the outside," analytical data is consumed by data analysts and data scientists to derive retrospective or future insights on the business. Thus, analytical data helps to optimize the business and user experience by using techniques like Machine Learning (ML) or business reports.

In short, analytical data provides information that supports the planning and decision-making activities of business management. The purpose of operational data is to collect information that supports activities related to running a business and apply it [19].

With both types of data already defined, storage options need to be discussed. Two appealing concepts are presented: Data Warehouses and Data Lakes.

A Data Warehouse is a specialized data management system created to facilitate and enhance business operations, particularly analytics, by centralizing and consolidating vast quantities of structured data from various origins. The primary purpose of Data Warehouses is to execute queries and conduct analyses, often encompassing extensive historical data. These warehouses typically accumulate information from diverse sources like application log files and transactional applications [20].

A Data Lake is similar to a Data Warehouse in the sense that it stores great quantities of data, with the biggest difference being how Data Warehouses store structured data, while Data Lakes store raw, unstructured, and semi-structured data in their native format. They adopt a schema-on-read approach, meaning that the data's structure is applied at the time of analysis or exploration. This allows for more flexible exploration and more advanced data analytics, at the cost of additional performance weight [21].

With the concept of Data Lake, a very interesting concept appears: the integration of Data Lakes with the Edge-Cloud Continuum technology. One way to achieve this integration would be with the Data Lake serving as a centralized storage and processing hub for data collected from various Edge devices and sensors. This would necessitate a process not unlike the following: First comes the data collection by IoT sensors, cameras, or industrial machinery, which generate data at the Edge of the network. This data at the Edge can be preprocessed or filtered using EC capabilities. This step helps filter out irrelevant or redundant data, allowing only valuable information to be sent to the Data Lake. Now the Data Lake receives the transmitted information and stores it for future use. This data can now be used for additional data processing or analytics, creating a useful resource [22].

3 Data Exchange

This section aims to resolve the question that arises regarding how data exchanges would occur, given the vision of the possibilities in data storage and the potential for combined integration with Edge-Cloud Continuum.

The exchange of vast amounts of data between Edge devices and the Cloud enables organizations to take advantage of the scalability and computational capabilities of the Cloud for complex data processing. The data exchange mechanisms should thus be secure, trusted, and privacy-protected for the information involved in the process. This is especially relevant in some application domains, such as intelligent transportation systems [23], industrial IoT [24], or healthcare [25]. Next, the DaaS paradigm involves a way of offering data to customers, enabling them to access, utilize, and analyze data without the need for complex infrastructure or expert knowledge in data management. With the Edge-Cloud Continuum, organizations can build data pipelines that collect and process data from Edge devices, aggregate it in the Cloud, and offer it as a service to customers. This data can be packaged in various formats, such as APIs, files, or data streams, allowing customers to integrate it into their own applications and analytics services. Finally, the utilization of these technologies allows organizations to provide real-time analytics and on-demand data access, empowering organizations to perform a secure exchange of information, make informed decisions, develop innovative applications, and extract maximum value from Edge-generated data.

3.1 Data Spaces

Many heterogeneous organizations collect and store a huge amount of data. The analysis of this data using ML techniques and Artificial Intelligence (AI) allows the organizations to optimize processes and activities. Moreover, new services and business cases can be created based on the use of ML and AI on massive volumes of data. Data ecosystems consist of a set of autonomous parties engaged in data sharing, which results in shared value creation and data-driven solutions. They are key to collaboration among different stakeholders. The data ecosystem parties want to be sure that their data will be used in an acceptable manner and that the data they receive from others has the expected characteristics. To facilitate the practical exchange of data, groups of parties can establish so-called Data Spaces. A Data Space offers the infrastructure that enables data transactions between different parties in a data ecosystem, operating under the specific Governance Framework of that Data Space (Fig. 2). The Governance Framework consists of a set of principles, standards, policies, and practices that apply to the governance, management, and operations [26].

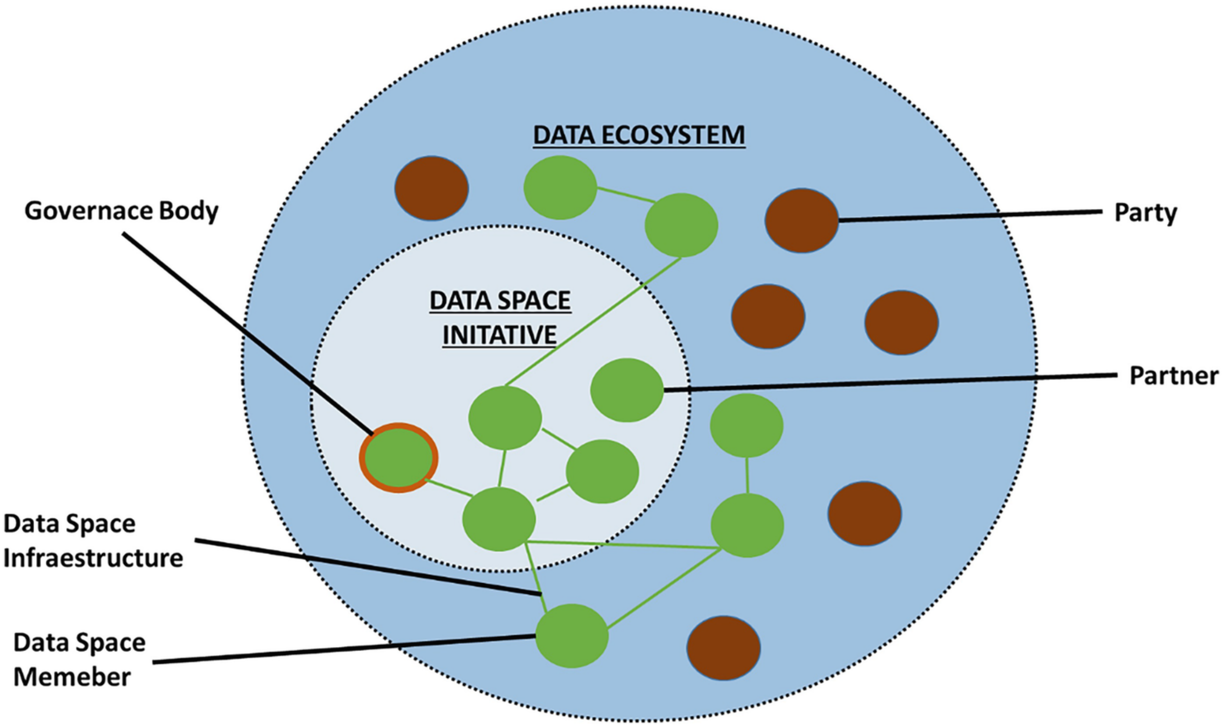


Fig. 2 Data spaces within the data ecosystem

The initiatives focused on deploying and maintaining a Data Space are known as Data Space Initiatives [3]. For instance, IDSA[1] and Gaia-X[2] are two of the currently existing Data Space Initiatives in Europe. Parties that commit to contributing to a Data Space Initiative are referred to as partners. A key responsibility of a Data Space.

Initiative is the development of a Governance Framework that encompasses the rules and practices governing the governance, management, and operations of the Data Space. The term “Governance Body” is used to describe the role played by a partner in the Data Space Initiative who is responsible for establishing and maintaining the Governance Framework.

Once the Data Space Governance Framework is established, various parties of the data ecosystem can become members of the Data Space if they meet the acceptance criteria. By joining a Data Space, members can benefit from the security and reliability offered by the framework while sharing data with other members within that Data Space. The Data Space Initiative proposes a set of enablers to support sharing data in a trusted way, and the data owners joining the Data Space determine how, when, and which way the data consumers can use this data, putting there the basis of the sovereignty in data sharing. Each Data Space provides domain-specific data, providing a solid basis for one or many data ecosystems [1].

The successful implementation of Data Spaces is based on the integration of various technological building blocks. First, the paramount importance lies in achieving data interoperability (Sect. 2.2), which enables efficient data exchange through the adoption of a shared language, standardized APIs, and common data models. Additionally, it is essential to incorporate mechanisms that ensure traceability and data provenance. Moreover, the establishment of data sovereignty and trust within Data Spaces becomes a priority, necessitating the implementation of uniform standards to manage participant identities, verify truthfulness, and enforce data access control policies. Lastly, Data Spaces should actively support the creation of value through multi-sided markets, which entails the adoption of standardized descriptions for terms, pricing, and data service offerings. Furthermore, these Data Spaces should enable effective publication, discovery, and accountability throughout the entire lifecycle of data service contracts.

To enable data sharing among participants, metadata provision is essential irrespective of the Data Space's design (centralized, federated, or decentralized) and the data's accessibility (open or protected). Publishing data with a standardized vocabulary for querying and implementing access controls for catalog items is necessary. While two participants can directly share data without a catalog, this function significantly enhances the discoverability of data assets and services when multiple participants are involved. In the case of a federated or decentralized design with multiple catalogs, the catalog service should support federated searches across various sites. It is important to note that catalogs do not host data assets themselves but provide data contract offers. When choosing an architecture for a Data Space, the catalog function's design can vary, spanning from a central catalog to multiple federated catalogs or numerous decentralized catalogs, based on the desired configuration along the spectrum.

The Data Space relies on data sovereignty principles aiming to achieve autonomy rather than complete independence, emphasizing the ability to make choices. It encompasses the control over data storage, determining where and when it is stored, and defining access protocols. In this regard, the concept aligns with the notion of privacy, as both emphasize the importance of individual control and decision-making. Furthermore, the realization of Data Spaces requires the integration of data governance elements that align with the business, operational, and organizational agreements of the participants involved. Business agreements play a crucial role in establishing the terms and conditions that govern data sharing and define the legal framework for contracts within the Data Space. Operational agreements are instrumental in enforcing policies such as compliance with regulatory frameworks like GDPR and may also include requirements for Cloud infrastructure operators and global service providers to implement robust auditing processes and cybersecurity practices. Lastly, organizational agreements assume responsibility for creating governance bodies that specify compliance standards for technology building blocks and make determinations regarding the adoption of business and operational agreements within the DS.

There are several key roles involved in data sharing (Fig. 3). Data Owners play a crucial role as entities that have ownership and control over specific datasets. They are responsible for determining how their data is shared and utilized within the DS. The Data Provider makes data available for exchange between a Data Owner and a Data Consumer. Data Providers offer data assets for sharing within the Data Space, making them available to other participants. The Data Space connectors serve as intermediaries in the Data Space architecture, facilitating secure and controlled data exchange between Data Providers and Data Consumers. They handle authentication, authorization, and data transfer, ensuring privacy and security during the process. The Data Consumers receive, consume, and process data acquired from the DS. Data Consumers typically employ data analytics, ML, or other technologies to extract insights or generate value from the shared data.

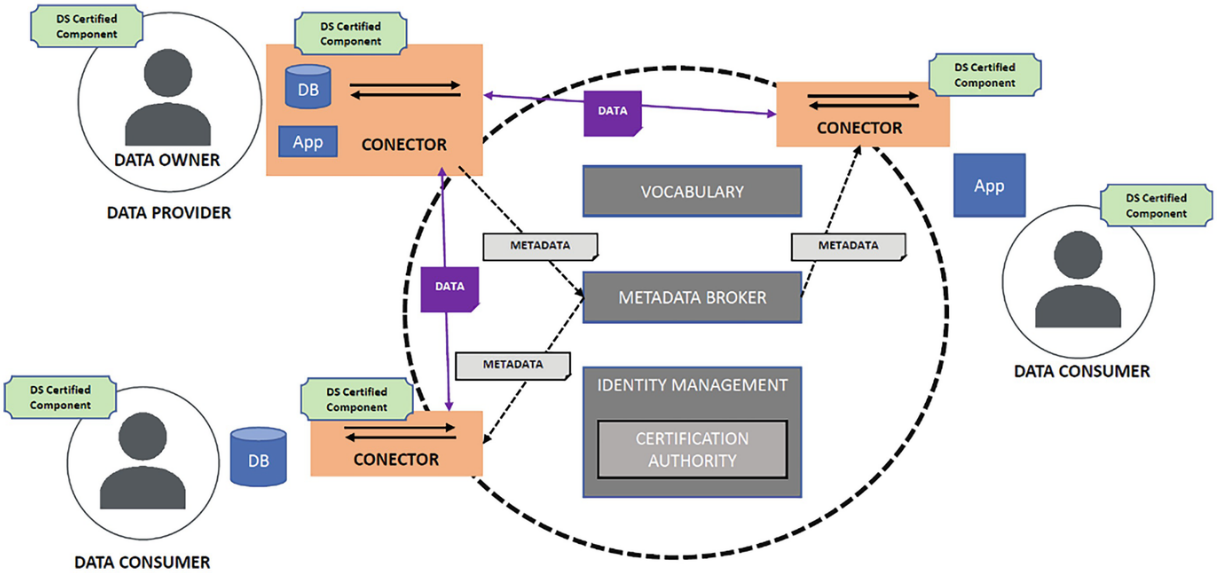

DS Certified Component
DATA OWNER
DATA PROVIDER
DB
App
CONECTOR
DATA
CONECTOR
App
VOCABULARY
METADATA
METADATA
METADATA BROKER
DATA
METADATA
IDENTITY MANAGEMENT
CERTIFICATION AUTHORITY
DATA CONSUMER
DB
CONECTOR
DATA CONSUMER

Fig. 3 Main actors in a Data Space

In addition, the process is complemented by other actors (Fig. 3). Trusted Third Parties (TTPs) play a significant role in establishing trust among participants. These independent entities offer various trust services, including identity verification, data certification, auditing, and compliance verification. By providing these services, TTPs contribute to the creation of trust within the Data Space framework. Certification Authorities (CAs) also contribute to trust-building by issuing digital certificates to participants. These certificates serve as proof of identity and ensure secure communication between participants. CAs verify participant identities and guarantee the integrity and authenticity of their digital credentials. Additionally, Data Space Brokers facilitate the process of data source discovery and matching within DS. They assist Data Users in finding relevant data providers and play a vital role in negotiating and establishing data sharing agreements between participants.

The IDSA initiative defines a framework for a secure and trusted environment for companies of all sizes and industries to manage their data assets within Data Spaces [27]. Specifically, the main concepts are defined in the IDS Functional Layer rulebook, providing the basis for enabling secure, trusted, and privacy-protected data exchange. The “Trust” group encompasses roles, identity management, and user certification, along with governance aspects. Each role in IDS has specific rights and responsibilities, with the Identity Provider playing a key role by offering services related to identity information. Identity management requires that every Connector in IDS possess a valid certificate and unique identifier, enabling the verification of other Connectors' identities. User certification is crucial for establishing trust among all participants, and it is explored further in the corresponding section. By addressing these trust-related aspects, the IDS Functional Layer ensures secure and reliable data exchange while maintaining privacy and confidentiality. The "Security and data sovereignty" group involves the following key features: authentication and authorization, trustworthy communication and security by design, usage policies and enforcement, and technical certification [27]. Authentication involves each Connector possessing a valid X.509 certificate, allowing participants to verify the identities of other participants. Authorization ensures that the Connector serving as the data source verifies the capabilities, security features, and identity of the receiving Connector. Usage policies and enforcement provide Data Owners and Providers with assurance that their data is handled according to specified policies, allowing participants to define restrictions and permissions for data handling. Trustworthy communication and security by design involve checking for trusted software stacks, encrypted and integrity-protected communication between Connectors, and adherence to usage policies. Technical certification is necessary for core components, such as Connectors, to establish trust among all participants, requiring certification from the designated Certification Body.

Finally, focusing on the infrastructure where data sharing occurs, the capabilities of Cloud and EC provide users with the flexibility to access a diverse range of resources for conducting computations or storing data in their preferred environment. The utilization of Data Spaces acts as the foundation for establishing a federation of the available data within these infrastructures. This enables smooth operation and seamless interconnection within the Data Spaces, facilitating secure and reliable data migration for users to either the Cloud or the Edge, based on their specific needs and preferences.

Next, a table showcasing the key points discussed in the section can be seen as Table 1.

Table 1 Data Spaces key points

Data Spaces key points
A key responsibility of a Data Space Initiative is the development of a Governance Framework that encompasses the rules and practices of the Data Space
The Data Space Initiative proposes a set of enablers to support sharing data in a trusted way, to enable data sharing among participants, metadata provision is essential irrespective of the Data Space’s design and data’s accessibility
The successful implementation of Data Spaces is based on data interoperability, traceability, data sovereignty, standardization and effective publication, discovery, and accountability
There are several key roles involved in data sharing: Data Owners, Data Providers and Data Consumers. The process is complemented by other actors such as Data Space Brokers, Trusted Third Parties and Certification Authorities
The capabilities of Cloud and Edge Computing provide users with the flexibility to access a diverse range of resources for conducting computations or storing data in their preferred environment. The utilization of Data Spaces acts as the foundation for establishing a federation of the available data within these infrastructures
The IDSA initiative defines a framework for data spaces, providing the basis for enabling secure, trusted, and privacy-protected data exchange:
1. The “trust” group encompasses roles, identity management, and user certification, along with governance aspects. User certification is crucial for establishing trust
2. The “security and data sovereignty” group involves the following key features: Authentication and authorization, trustworthy communication and security by design, usage policies and enforcement, and technical certification
The utilization of Data Spaces acts as the foundation for establishing a federation of the available data within Cloud and Edge Computing intensive infrastructures

3.2 Data as a Service

DaaS is a data management strategy that relies on the creation of a service-based model to access the data, where a series of distributed services cover all the steps of the data lifecycle (from data collection to its delivery to the data consumer). These services expose APIs to enable access to the data assets. This way, the data can be served with different levels of granularity, depending on the specific needs of each user. DaaS can be seen as an implementation of Service-oriented Architecture (SOA) because DaaS applications are typically built using service-oriented principles. The term has been in use for over a decade, and it is difficult to pinpoint a specific date or source for its initial definition. One early mention of DaaS can be found in Rajesh et al. [28] where they define DaaS as follows: “*Data as a service, or DaaS, is a cousin of software as a service. Like all members of the ‘as a Service’ (aaS) family, DaaS is based on the concept that the* *product, data in this case, can be provided on demand to the user regardless of geographic or organizational separation of provider and consumer.*" Originally rooted in Cloud-based solutions, DaaS is witnessing a transformation with the emergence of Fog Computing, extending the capabilities of CC towards the edge of the network.

DaaS introduces the concept of centralizing data quality, where data is cleaned and enriched in a single location and then made available to various systems, applications, or users regardless of their location within the organization or network. Consequently, DaaS solutions offer the following benefits:

- *Data Quality*: Control over data access is managed through the utilization of data services, resulting in enhanced data quality as updates are consolidated at a single point. After thorough testing of these services, the need for future regression testing is minimized unless any modifications are made during subsequent deployments.
- *Agility*: Consumers can swiftly adapt and take action thanks to the simplicity of data access, without the need for extensive knowledge of the underlying data. If consumers require a slightly different data structure or have location-specific requirements, the implementation process is uncomplicated due to the minimal changes involved.
- *Cost-Effectiveness*: Providers have the opportunity to construct the foundation using data experts and delegate the development of the presentation layer to external sources. This approach leads to highly cost-effective user interfaces and enhances the practicality of implementing change requests at the presentation layer.

Although DaaS is scalable and eases access to the data regardless of the consumer's location, it has significant limitations. First, given the heterogeneity of the data delivered, it is necessary to handle different data sources with different types of data (i.e., content, schema, or delivery type). However, currently, there is no standard way to define those services or the supplied data due to the lack of a global machine-readable information model for DaaS. This lack of standards hinders interoperability and often results in poor metadata quality, which limits the ability of the users to find and select the most suitable datasets and extract useful insights from them. Moreover, the original Cloud approach causes significant latencies, and moving the data from the Edge to the Cloud raises security and privacy concerns. Also, the intellectual property implications must be considered in each case. Due to the previous points, service composition is not straightforward. Furthermore, the data provided must meet the quality requirements of the users. To that aim, Quality of Service (QoS) and Quality of Data (QoD) monitoring and evaluation mechanisms are needed. Additionally, DaaS presents limited customization because it typically provides standardized data sets that may not fully meet the specific needs of every user.

Another concept related to DaaS is the Data Marketplace (or DaaS Marketplace), which enables data sharing among organizations or individuals by acting as a mediator and should be able to enforce the pertinent rules, laws, and regulations. In addition to facilitating the data exchange and the corresponding payments, Data Marketplaces usually provide other functionalities that aid in this process, such as data cataloging, data discovery, or metadata management [29]. It is worth noting that having rich metadata descriptions of the data is crucial to be able to provide data cataloging, search and discovery services, as well as to allow an effective use of the data. H2020 Project TRUSTS [29] identified and classified 35 functionalities associated with Data Marketplaces. Some of the most relevant categories among those were "trading arrangements," "data transformation," "dataset discovery," "data governance," and "user management." Regarding their topologies, Data Marketplaces can have a centralized approach based on a single infrastructure that hosts and processes all the data and metadata, or a federated approach. Data Space initiatives such as IDSA or Gaia-X (Sect. 3.1) promote the creation of federated data marketplaces.

Although many Data Marketplaces have emerged in the last few years, they still face numerous challenges that need to be addressed to ensure their effectiveness and sustainability:

- *Data Quality and Integrity*: Data originates from diverse sources, each utilizing distinct methods of collection, standards, and levels of precision. Data Marketplaces should ensure that the provided data meets data quality and integrity requirements.
- *Data Governance*: It is essential to establish clear rules and guidelines for data usage, intellectual property rights, and ownership. To ensure that data consumers do not violate data access and usage restrictions, data

providers should be able to track down data usage and ensure compliance with data sharing agreements.

- *Data Privacy and Security*: Data Marketplaces handle highly sensitive and confidential data. Protecting personal and proprietary information from unauthorized access, breaches, or misuse is of major importance.
- *Lack of Transparency*: Transparency is a key element for building trust and confidence among data providers and buyers. It enables them to make well-informed decisions by providing visibility into the operations and processes involved. Data Marketplaces should provide clear information about data sources, pricing, collection methods, pricing, security measures, and usage guidelines.
- *Efficiency and Scalability*: Data Marketplaces should ensure smooth and timely transactions by facilitating seamless data integration and implementing efficient data processing and analytics workflows. Moreover, they should be built on top of a flexible architecture that can adapt to a growing demand.

Subsequently, in Table 2, the key points of the section are abridged:

Table 2 Data as a Service key points

Data as a Service key points
DaaS is a data management strategy which relies on the creation of a service-based model to access the data, where a series of distributed services cover all the steps of the data lifecycle. This way, the data can be served with different levels of granularity, depending on the specific needs of each user
DaaS solutions offer the following benefits:
1. Data quality
2. Agility
3. Cost-effectiveness
However, it has significant limitations:
1. Lack of standards hindering interoperability and causing poor metadata quality
2. Significant latency issues in the original Cloud approach
3. Intellectual property implications
4. To combat this, Quality of Service and Quality of Data mechanisms are needed.
A concept related to DaaS is the Data Marketplace, which enables data sharing among organizations or individuals by acting as a mediator and should be able to enforce the pertinent rules, laws, and regulations. Data Space initiatives such as IDSA or Gaia-X promote the creation of federated data marketplaces
The emerging Data Marketplaces still face numerous challenges that need to be addressed:

Data as a Service key points
1. Data quality and integrity
2. Data governance
3. Data privacy and Security
4. Lack of transparency
5. Efficiency and scalability

4 Modern Data Management Frameworks

Continuing with the DaaS approach explained in the previous section, this section builds upon DaaS and goes beyond it, proposing DaaP as the core enabling concept to facilitate the implementation of digital continuity. This section will focus on delving and explaining state-of-the-art data management frameworks, more specifically the noteworthy Data Fabric and Data Mesh, which are still novel concepts more focused on both the commercial and business sectors. The section also aims to explain how both concepts, despite some differences, have several points in common and could provide relevant synergies between them. Explaining the DaaP approach proposed and introducing the basis for the chapter’s proposed use case is one of the main focuses of the section. After every subsection, a table will be presented, highlighting the contents of the pertinent subsection in the forms of Tables 3, 4, 5, and 6.

Table 3 Data Fabric key points

Data Fabric key points
The Data Fabric is a technical architecture that aims to connect processes and dynamically integrate data within an organization, thus enabling seamless data movement and use across the organization in a flexible and reusable way
It makes use of data virtualization combined with a rich metadata representation of the knowledge to offer access to the data to the users of the organization
Data virtualization is based on the creation of an abstraction layer that defines a common way to access the data through a unified interface while keeping it at the sources
All the available data assets should be registered in the Data Catalog to make them searchable and discoverable. The Knowledge Graph describes the connections among the distributed data sources and services (which constitute the nodes) using semantically linked metadata
The Data Fabric concept was initially associated with Cloud or multi-cloud environments, but in recent years its scope is being expanded to integrate new environments such as EC or industrial

Data Fabric key points
IoT, as well as to promote the inclusion of AI/ML techniques to increase its automation capabilities

Table 4 Data as a Product key points

Data as a Product key points
In Data Economy, Data Product is understood as a collection of data that has been packaged and made ready for data trading in a marketplace and is properly described using metadata. Another definition is Gaia-X's, where a Data Product contains the actual data that is shared and its metadata description (Self-Description)
Concerns arise regarding the usefulness, usability, and quality of the data. For this the DaaP solution is presented: The basis of DaaP is the application of product thinking to the data assets of an organization. This way, DaaP considers the entire process of preparing the data to be served to the consumers
To fulfill the DaaP requisites, the data must have the following basic qualities:
1. Discoverable
2. Addressable
3. Trustworthy and truthful
4. Self-describing semantics and syntax
5. Interoperable
6. Secure

Table 5 Data Mesh key points

Data Mesh key points
The Data Mesh is a socio-technical concept that aims to create a domain-oriented decentralized architecture to extract and manage high amounts of analytical data in large-scale environments within or across organizations
The Data Mesh definition is based on four key principles:
1. Domain-oriented decentralized data ownership and architecture
2. Data as a Product
3. Self-serve data infrastructure as a platform
4. Federated computational governance.
The adoption of the Data Mesh approach has several potential benefits:
1. Easily scalable architecture
2. Facilitation of the definition and enforcement of data access policies
3. Collaboration between business and IT departments
4. Speeds up data delivery

Data Mesh key points
5. High-quality self-service data outputs
It also has potential limitations:
1. Hard and challenging adoption
2. Customized for each use-case
3. The adoption needs a tactical and strategic approach to be efficient
4. It can create data-silos

Table 6 Synergies between modern Data Management approaches key points

Synergies between modern Data Management approaches key points
Both the Data Mesh and the Data Fabric aim at easing the management of huge amounts of data and enabling the integration of heterogeneous data sources
They can both be seen as complimentary; Data Fabric provides a set of technical capabilities that can be the basis to build a Data Mesh solution:
1. The Knowledge Graph can describe relationships between elements
2. Metadata can help simplify and automatize tasks in the Data Mesh
3. The data cataloguing and discovery of Data Fabric enables some of the required functionalities for Data Products in the Data Mesh
4. The automation and self-service capabilities of the Data Fabric can set the basis to build the self-serve data platform of the Data Mesh

4.1 Data Fabric

The Data Fabric is a technical architecture that aims to connect processes and dynamically integrate data within an organization, thus enabling seamless data movement and use across the organization in a flexible and reusable way. It promotes a data-centric mindset, which means that the data consumers do not need to deal with the burden of searching for and collecting data from heterogeneous sources. With the Data Fabric, they should be able to simply ask for the data they need. Furthermore, since the Data Fabric separates the data usage from the data infrastructure, it allows for new ways of using the data that were not foreseen initially. The Data Fabric applies knowledge-based integration techniques to provide a unified view of the data regardless of the location and heterogeneity of the sources [30]. More concretely, it makes use of data virtualization combined with a rich metadata representation of the knowledge to offer access to the data to the users of the organization.

Data virtualization is based on the creation of an abstraction layer that defines a common way to access the data through a unified interface while keeping it at the source (whether on-premises, Edge, or Cloud). It is important to note that in the Data Fabric not all the data is virtualized due to the extra latency that virtualization adds to the process (when compared to the use of a centralized data storage). Thus, which data should be virtualized is decided in each implementation of the Data Fabric.

Regarding the metadata, it comprises a Data Catalog and a Knowledge Graph. All the available data assets should be registered in the Data Catalog in order to make them searchable and discoverable. The Knowledge Graph describes the connections among the distributed data sources and services (which constitute the nodes) using semantically linked metadata, thus providing a structured semantic representation of the data in the business context. Hence, the Knowledge Graph allows understanding and integrating data from distributed sources. According to Gartner [31, 32], in the Data Fabric, metadata should be continuously managed, analyzed, and augmented ("active metadata") by making use of ML and AI. Data governance also plays a key role in the Data Fabric architecture.

Although the Data Fabric defines a technical architecture and a set of principles, its implementation must consider the specific needs of each use case. As a result, the Data Fabric implementations may vary significantly. Moreover, the Data Fabric concept was initially associated with Cloud or multi-cloud environments, but in recent years, its scope is being expanded to integrate new environments such as EC or industrial IoT, as well as to promote the inclusion of AI/ML techniques to increase its automation capabilities [33].

4.2 Data as a Product

In Data Economy, a Data Product is understood as a collection of data that has been packaged and made ready for data trading in a marketplace and is properly described using metadata. These data products can be derived directly from data assets or be the result of the analysis or combination of primitive data [34].

A similar definition of Data Product can be found in Gaia-X, where a Data Product contains the actual data that is shared and its metadata description (Self-Description). This Self-Description contains a license that describes the terms and conditions associated with the data and can be used to initiate a contract negotiation between the data consumer and the data provider. The Self-Description is stored in a federated Data Catalog to make it available to all participants (Sect. 3.1).

However, some concerns arise regarding the usefulness, usability, and quality of the data, as well as the ability to combine different data assets in order to extract more valuable insights. To address these common issues, Zhamak Dehghani [35] proposed the DaaP approach. The basis of DaaP is the application of product thinking to the data assets of an organization. This way, the consumers of the data are seen as customers whose needs must be fulfilled, and, as a result, the availability of quality analytical data becomes a priority. Hence, DaaP considers the entire process of preparing the data to be served to the consumers, including aspects such as data quality and decreased lead time for data consumption. According to Dehghani, to meet the needs of the data consumers, the data must have the following basic qualities:

- *Discoverable*: The consumers should be able to find interesting Data Products easily. A common way to achieve this is to define a discovery service based on a registry that contains the metadata of all the available datasets and is usually centralized.
- *Addressable*: Each Data Product must have a unique address (endpoint) to ensure that data consumers can access the data programmatically. In a decentralized architecture, these addresses must follow a defined global convention to facilitate the task of finding and accessing the data.
- *Trustworthy* and *Truthful*: The data must reflect the reality of the organization to be useful. Some of the techniques that can be applied to ensure this are data cleansing and data quality assessment, as well as providing data provenance and data lineage information as part of the metadata.
- *Self-describing Semantics and Syntax*: The data consumers must be able to access and use the data without direct intervention of the data provider. To achieve this, the use of well-defined syntax and semantics is crucial. Metadata also plays an important role in the process of making the data understandable, as well as in the definition of the data quality objectives and data usage policies.
- *Interoperable*: The existence and use of standards and harmonization rules are necessary to allow data sharing and correlation across different domains because naming conventions, attributes, metadata fields, and

identifiers may be different in each domain. Hence, standardization allows the data consumers to correlate and combine data from different domains in a meaningful way.
- *Secure*: In order to be able to share data in a secure way, data confidentiality, availability, and integrity must be ensured. Data access control is necessary, with access control policies defined with the appropriate granularity for each data product.

Following this approach, a Data Product comprises not only the data and metadata but also the code to provide access to the data following the defined access policies and the necessary infrastructure to run this code. Hence, this concept differs from the usual Data Product definition, where only the data and metadata are considered.

4.3 Data Mesh

The concept of "Data Mesh" was defined by Zhamak Dehghani in 2019 [35]. The Data Mesh is a socio-technical concept that aims to create a domain-oriented, decentralized architecture to extract and manage large amounts of analytical data in large-scale environments within or across organizations. This model arises as a way to solve the problems associated with the proliferation of data sources, such as the high cost of the management of monolithic or centralized data architectures, the difficulties associated with the use of the data by the data teams, etc. This implies making changes not only at the technological level, but also at the organizational level. The Data Mesh divides the analytical data into a set of data domains, which are aligned with the business domains, and shifts the responsibility of managing the data to domain teams, who have a better understanding of the data than a central data team. The Data Mesh definition is based on four key principles [36]:

- *Domain-Oriented Decentralized Data Ownership and Architecture*: In order to enable scalable data sharing, data ownership becomes decentralized. Hence, the data architecture should be modeled according to the different business domains of the organization, and each business domain should be responsible for storing and serving their data.
- *Data as a Product*: As has been described in the previous section, data is understood and managed as an asset to be shared across different domains. The different data products are built and managed by the corresponding domains and constitute the nodes of the data mesh. Hence,

in the Data Mesh, the data, code, and infrastructure are not managed independently but as a unit where the focus is on the data itself.
- *Self-Serve Data Infrastructure as a Platform*: To enable a decentralized management of the data, the self-serve data platform must provide the tools to create, maintain, and serve the Data Products with less specialized knowledge than with the current technologies. This implies hiding the complexity associated with building and sharing the Data Products, as well as the use of automation where possible. From the data consumer's side, the platform must enable seamless discovery and use of the data.
- *Federated Computational Governance*: To allow interoperability among the different domains as well as decentralization, the Data Mesh requires a decentralized approach to data governance. This governance model must be able to support a dynamic topology as well as interoperability based on global standards and allow the independence of each domain in a decentralized environment. In addition, it must enable the automated execution of decisions by the platform. In order to be able to share and combine Data Products across the data mesh, the coexistence of a set of global governance rules and local decision-making power within the domains is needed. Thus, this approach to data governance requires a balance between decentralization and centralization.

Although the concept of "Data Mesh" is still relatively new, it has gained popularity since it was coined. Recent studies [37, 38] have analyzed the available information on the topic in order to consolidate the concept and identify possible approaches for its design and implementation, as well as the main challenges that must be tackled. Based on this analysis, the first reference architectures for the Data Mesh have been designed. Butte and Butte [39] proposed a generic reference architecture for the Data Mesh and described how it could be implemented in Amazon Web Services cloud. Another reference architecture was proposed by Goedegebuure et al. [38], who mapped the Data Mesh capabilities with the SOA model.

The adoption of the Data Mesh approach has several potential benefits [40]: (i) its decentralized architecture is easily scalable and can improve the business domain agility, (ii) the management of the data on a domain basis coupled with federated data governance facilitates the definition and enforcement of data access policies, (iii) it promotes collaboration between business and IT departments, which results in increased transparency, (iv) it speeds up data delivery while hiding the underlying complexity, and (v) it provides high-quality self-service data outputs. However, the Data Mesh also has some potential limitations: (i) the shift of responsibility towards the domains can make its adoption and management challenging, (ii) it has to be customized for each use case, (iii) its adoption needs a strategic and tactical approach to be efficient, and (iv) the domain-oriented focus of the Data Mesh has the potential to create data silos.

Bode et al. [41] conducted semi-structured interviews with industry experts in order to gain some insights into the main challenges associated with the Data Mesh. The main challenges that they identified are the following:

- The transition toward a federated data governance model and its associated security, privacy, and regulatory considerations.
- The provision of adequate metadata descriptions and proper data modeling, which are necessary to enable data product sharing among different domains.
- The shift of responsibility toward the domains can be hindered by a lack of resources, competences, or motivation.
- Limited technical, human, or financial resources.
- Acceptance and resistance issues within the organizations.
- Lack of comprehension of the Data Mesh concept in the organizations that may lead to an improper use of the term instead of an actual adoption of the model.

4.4 Synergies Between Modern Data Management Approaches

From the definitions provided in the previous sections, it is clear that the Data Mesh and the Data Fabric have the same goals. Both concepts aim at easing the management of huge amounts of data and enabling the integration of heterogeneous data sources. However, these concepts are different: the Data Fabric defines a technical architecture and relies purely on technical capabilities such as automated metadata or data profiling, whereas the Data Mesh is a socio-technical concept and, as such, it also has implications at organizational level (i.e., shift of data management responsibilities towards the domain teams). As depicted in Fig. 4, both concepts share certain similarities while differing in other aspects. From a technical point of view, the main differences are the following:

- In a Data Fabric, the approach to data exchange basically follows the ideas described in Sect. 3 and improves them by adding automation and knowledge-based data integration techniques. Although Data Fabric makes use of data virtualization, not all the data is virtualized. In contrast, the Data Mesh is based on the DaaP approach proposed by Dehghani, which goes one step further by addressing common issues associated with DaaS and Data Marketplaces.
- Federated data governance is one of the key principles of the Data Mesh, whereas in Data Fabric data governance is typically centralized. Related to this is data ownership, which in the Data Mesh is defined by domain, whereas in Data Fabric there is no clear general definition.
- Data Fabric defines a unified security model. In the Data Mesh, the security is defined and applied on a domain basis.
- The Data Fabric provides automatic mechanisms for data consistency. In the Data Mesh, achieving data consistency is more complex due to its decentralized approach.

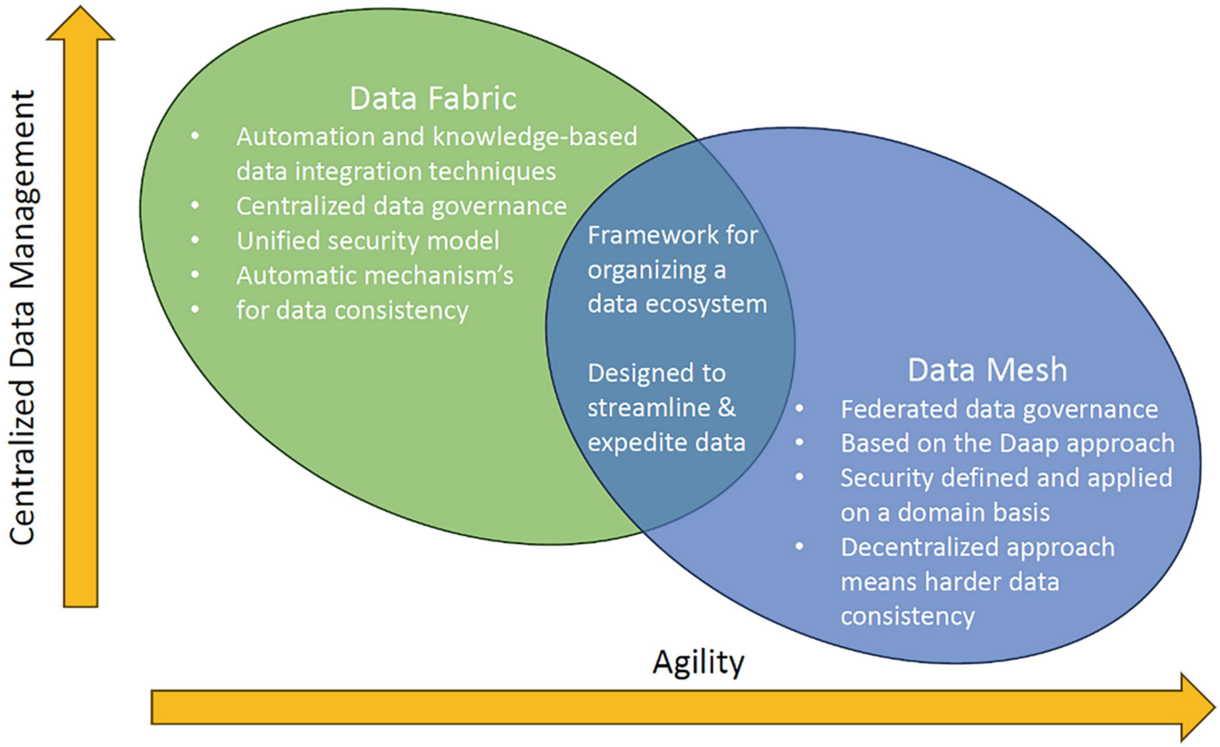

Centralized Data Management
Data Fabric
• Automation and knowledge-based data integration techniques
• Centralized data governance
• Unified security model
• Automatic mechanism's
• for data consistency
Framework for organizing a data ecosystem
Designed to streamline & expedite data
Data Mesh
• Federated data governance
• Based on the Daap approach
• Security defined and applied on a domain basis
• Decentralized approach means harder data consistency
Agility

Fig. 4 Synergies between Data Fabric and Data Mesh

To sum up, from a technical point of view, the implementation of the Data Mesh is more challenging due to its decentralized nature.

Although different, Data Fabric and Data Mesh can be understood as complementary concepts. The Data Fabric provides a set of technical capabilities that can be the basis to build a Data Mesh solution [42]:

- The *Knowledge Graph* can describe the relationships among data, systems, processes, and people within the organization.
- The use of *active metadata* can help simplify and automatize tasks in the Data Mesh.
- The *data cataloging and discovery* provided by the Data Fabric (supported by the use of metadata) enable some of the required functionalities for building, managing, and consuming Data Products in the Data Mesh. However, the federated approach of the Data Mesh makes necessary the use of orchestration mechanisms among data catalogues.
- The *automation and self-service capabilities* of the Data Fabric, such as the semantic search and the creation and management of active metadata, can set the basis to build the self-serve data platform of the Data Mesh.

5 Use Case: RE4DY Project

In the context of the green, circular, and digital transformation of the European manufacturing community, it is of critical importance to integrate innovative and proactive resilience strategies into data-driven digital manufacturing processes. These strategies should be implemented at both the production and supply chain levels, ensuring the preservation of sovereignty and competitiveness while upholding European digital values such as excellence, privacy, and trust. The objective is to enhance individual and value chain flexibility, thereby achieving long-term resilience and the ability to swiftly respond to any disruptions by reorganizing supply chains or expediting decision-making processes. To accomplish this, it is imperative to establish distributed data-intensive, intelligent, and dynamic industrial decision support systems. These systems should incorporate AI to enable smart anticipation and intelligent automation to facilitate rapid responses. The collaboration between humans and automation, known as Human-Automation symbiosis, is crucial in this endeavor. Only businesses that can effectively leverage their data, utilizing AI, digital threads, and Digital Twin solutions, will possess the agility to promptly address external shocks. This understanding of the digital thread concept refers to the use of digital tools and representations for design, evaluation, and life cycle management.

The Horizon Europe Project RE4DY proposes DaaP as a core concept for the implementation of digital continuity across Data Spaces, digital threads, ML pipelines, and Digital Twin workflows across all phases of the product and process lifecycle. The project aims to build data-driven digital value networks to build active resilience strategies and yield competitive advantages to the European industry through the integration of AI and Digital Twin across all phases of the processes. By leveraging DaaP, resilience can be enhanced alongside advanced manufacturing digital processes and value ecosystems. Additionally, the implementation of distributed data management solutions, aimed at bolstering factory resiliency, can be seamlessly repurposed to enhance connected factory and value network-level processes.

5.1 Technical Approach

The RE4DY concept emphasizes the adoption of modern distributed data management and integration architectures, which are rooted in Data Fabric design principles. Those architectures should be extended to encompass value ecosystems and Data Spaces. In order to effectively address the growing interconnectivity and autonomy of manufacturing value networks, it is essential to integrate these principles with Data Space sovereignty principles and Digital Twins. Moreover, RE4DY aims to include the aforementioned concepts, frameworks, and models (Sects. 2, 3, and 4) to create a trusted, interoperable, and low-cost manufacturing data network. By doing so, RE4DY will establish a framework for production value ecosystems in connected smart factories (Fig. 5) to facilitate the implementation of Cognitive Digital Twins and industrial Data Spaces (following the principles of IDSA and Gaia-X) and accelerate the adoption of integrated intelligence in smart manufacturing.

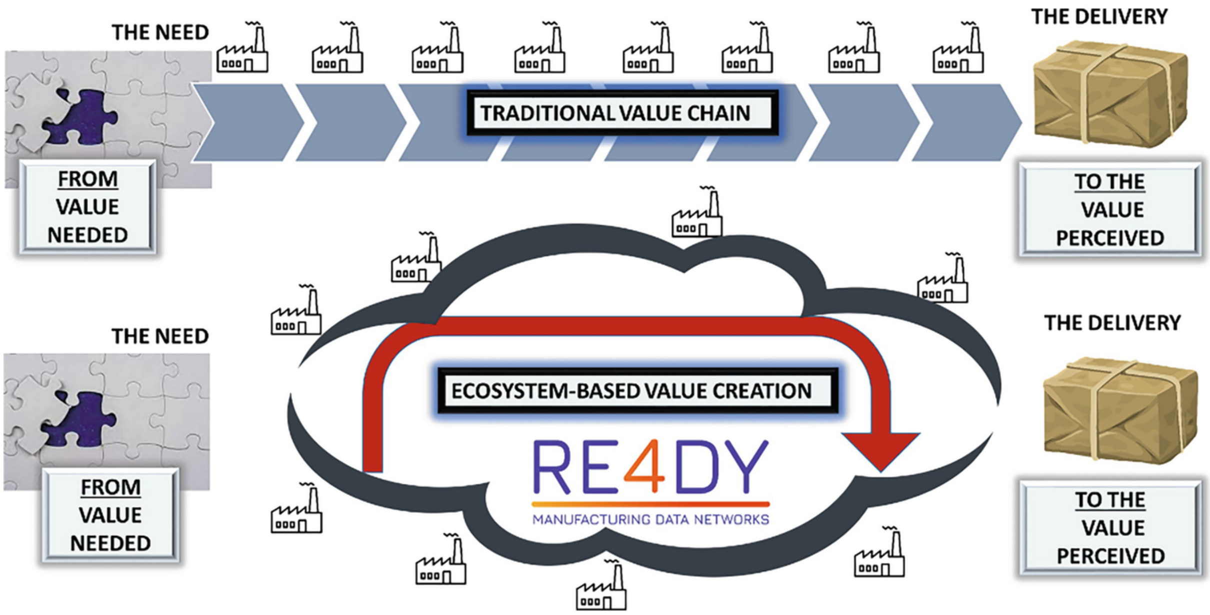

THE NEED
FROM VALUE NEEDED
TRADITIONAL VALUE CHAIN
THE DELIVERY
TO THE VALUE PERCEIVED
THE NEED
FROM VALUE NEEDED
ECOSYSTEM-BASED VALUE CREATION
RE4DY
MANUFACTURING DATA NETWORKS
THE DELIVERY
TO THE VALUE PERCEIVED

Fig. 5 RE4DY concept

In terms of infrastructures, RE4DY is committed to a decentralized model. Decentralization involves the deployment and management of resources across multiple physical or virtual locations. This distribution of infrastructure components enables better resource utilization, load balancing, and fault isolation, thereby increasing the overall efficiency and reliability of the system. Reinforcing decentralization on infrastructures, services, and data planes not only enhances fault tolerance and robustness, but also facilitates collaboration and interoperability among federated entities, e.g., exchanging data as a valuable product. Furthermore, decentralization of the data plane involves distributing data storage and processing capabilities across multiple nodes or Edge devices. This not only reduces the reliance on a central data repository but also improves data availability and responsiveness as it enables data to be processed and analyzed closer to the source, namely at Edge level, minimizing latency.

The adoption of the DaaP approach will enable a controlled and secure way to share meaningful and actionable data, which will be supported by Digital Continuum technologies. Based on this approach, RE4DY will develop a toolkit and services that will allow the creation and management of distributed data value ecosystems.

RE4DY strives to deliver services for Data Transformation and Data Virtualization as they are core technologies that enable a Data Fabric approach. The project’s objective is to connect to the different sources, integrating only the metadata required and creating a virtual data layer. This enables users to leverage the source data in real time. The RE4DY Data Fabric will prepare the data automatically and make it available for the ML/AI models while enhancing reusability and sustainability through the use of shared data models and automated data orchestration. In this respect, the definition of Knowledge Graphs and active metadata will enhance the data integration and provisioning capabilities of the RE4DY toolkit. Moreover, since Knowledge Graphs define a structured representation of the existing entities and services, their use can be leveraged in decision-making during lifecycle management. RE4DY will also implement the necessary self-service marketplace functionalities to support the creation and management of Data Products, as well as specific data contracts for trusted data sharing. The Digital Continuum will provide seamless access to the relevant data to the different stakeholders involved in the manufacturing process, as well as allow them to share their own data. This collaboration could lead to faster innovation and decision-making, as well as shorter product development cycles. In addition, the digitization of the value chain combined with the DaaP model will enable the collection and analysis of vast amounts of data across the manufacturing process that could provide valuable insights or enable the creation of predictive models or data-driven consulting services. Another possibility is the exploitation of the Data Products as an asset that can be monetized.

RE4DY seeks to boost the autonomy and interoperability of Big Data pipelines, digital thread and Digital Twins and promote an open, trusted, and comprehensive data governance. Moreover, RE4DY will build an integrated framework for the continuous delivery of industrial AI services and Cognitive Digital Twins and define a legal and methodological resiliency engineering framework. This will provide the means for the implementation of distributed value network processes along the full product lifecycle. As a result, manufacturers could be able to reduce defects, optimize processes, and increase product quality by making decisions based on the insights obtained from the analysis of the data.

5.2 Pilots

The outcomes of RE4DY will be validated in four pilots, which address both machine tool builder and product manufacturing resiliency perspectives, as well as joint optimization of smart product and production.

These pilots will create supply and manufacturing value network ecosystems in four strategic sectors, namely, automotive, aeronautics, eBattery, and machine tools and consumables.

- *Automotive: Connected Resilient Logistics Design and Planning in Automotive*: Volkswagen Autoeuropa[3] is a brownfield plant driven by low-cost, practical, and simple operational solutions to accommodate frantic production uptimes. The Volkswagen group has a growing emphasis on developing solutions for the logistics processes through digitalization and data analytics. RE4DY will permit making the data of each production plant easily accessible to everyone, through fluid user experience, enabling and empowering logistics planning specialists with the necessary information to generate key insights about the logistics processes. The integration of the RE4DY framework offers the following benefits: (i) Autonomous/Automatic Planning, (ii) lead time reduction, (iii) overall efficiency increase, (iv) increase in process flexibility, and (v) operating cost reduction.
- *Aeronautics: Cooperative Multi-plant Turbine Production with Predictive Quality Chains*: Avio Aero[4] is an aviation business specialized in the design, manufacturing, and maintenance of civil and military aeronautics subsystems and systems. The integration with the RE4DY Framework aims to establish a platform for near real-time predictive quality in distributed multi-plant aeronautical manufacturing processes. This integration is built upon four main pillars, which include: (i) a defect detection tools based on AI/ML to support quality inspection operations, (ii) predictive quality algorithms developed for a family of products and able to implement CIP processes, (iii) a new approach to speed up training of certified inspection operators and soften the strong dependency on senior certified operators, leveraging the knowledge base created with the tool based on AI/ML, (iv) a manufacturing digital thread designed to enable an effective data sharing and consumption in a multi-Digital Twin environment, geographically distributed over multiple plants and ready for the extension to additional value chain players.
- *E-Battery: Collaborative Ecosystem Resilient Product/Production System Engineering*: The Battery Innovation Center[5] was established to address several objectives in the field of battery technology. It focuses on developing battery modules and optimizing their production or recycling

processes in a near-series environment. The integration with READY enables the following benefits: (i) developing a comprehensive model for optimizing industrial process efficiency, sustainability, and reliability across individual operations and value chains, (ii) integrating distributed data management operations to enhance data reusability, portability, and pipeline efficiency, (iii) improving the efficiency, scalability, and trustworthiness of distributed Industrial Internet value networks, including data sharing capabilities, and (iv) facilitating access to data sets and supporting the establishment of resilient digital twin processes, empowering organizations to become trusted contributors within value networks.
- *Machine Tools and Consumables: Coordinated Ecosystem Integrated Performance Self-optimization*: The Swiss Smart Factory[6] is an extensive test and demo laboratory for Industry 4.0. It consists of an ecosystem that includes over 70 companies dedicated to performing and supporting industry-focused applied research and development. The adoption of the RE4DY Framework will allow cross-company data sharing in multi-cloud environments. This leads to more complete and integrated data sources, which ultimately leads to better decision-making with clear benefits for the end-users: (i) increased reliability of machining processes, (ii) increased transparency and autonomy of machining and supply chain processes, (iii) reduced energy consumption during machining processes, (iv) increased diagnostic and predictive capabilities.

---

6 Conclusions and Future Work

The Digital Continuum emerges from the combination of Edge and Cloud resources. The exchange of vast amounts of data between the Edge and the Cloud enables organizations to fully leverage the scalability and computational capabilities of the Cloud for complex processing. In this way, data pipelines can be built to collect and process data from the Edge, aggregate it in the Cloud, and then offer the service to customers. By capitalizing on the establishment of the Continuum, the focus is on providing interoperable data that enables efficient data exchange and correlation across different domains.

As the years go on, the rising necessity for data management has only grown, just like the importance of ML and AI technologies. It is for these reasons that new solutions pushing for better data exploitation have started to appear. They are focused on building unique data-driven digital value networks, sustaining competitive advantages through digital continuity and sovereign Data Spaces across all phases of a process and product life cycle. These approaches need a base to stand on like Edge-Cloud Continuum, data lifecycle and Data Exchange. The usage of Data Spaces acts as the foundation for establishing a federation of the available data within these infrastructures, thus enabling smooth interoperability and interconnection, facilitating secure and reliable data migration for users to either the Cloud or the Edge. Looking ahead, the current work is focused on refining and evaluating the existing data exchange solutions, facilitating interoperability, guaranteeing trust, managing data governance, offering DaaS, and ensuring compliance with privacy regulations. The mentioned DaaS approach has revolutionized data quality by centralizing its management, ensuring that data is meticulously cleansed and enhanced in a unified hub. By consolidating these processes, organizations can ensure that the data accessible to various systems, applications, and users is of the highest quality, regardless of their location within the organization or network. These approaches offer valuable features and advantages that can be exploited to overcome the complexities of data exchange in the Edge-Cloud Continuum.

Building upon the established approach of Data Spaces and DaaS, this chapter takes a further step by proposing DaaP as the key concept to lay the foundation for modern solutions based on digital continuity. This way distributed data management solutions can be seamlessly reutilized through connected and value network processes. Data Fabric and Data Mesh are innovative frameworks that align with the concept of DaaP, providing a robust foundation for managing and leveraging data in modern digital continuity solutions. These two modern approaches provide a framework for data management focused on both commercial and business sectors. Since both concepts aim at easing the management of huge amounts of data and enabling the integration of heterogeneous data sources, they overlap with one another, and they are now used to build one on top of the other. The Data Fabric provides a set of technical capabilities that are used to build the Data Mesh solution, creating a decentralized architecture that manages huge amounts of analytical data, thus overcoming the limitations of traditional centralized data management approaches and enabling organizations to leverage the full potential of their data assets. It provides data quality and data democratization, fostering a culture of data-driven decision-making.

The apex of these topics is the RE4DY initiative. The RE4DY Digital 4.0 Continuum technologies allow data streams to flow on a distributed industrial value network in a secure manner, with each sender or receiver sharing data in a DaaP context, making it actionable and meaningful. RE4DY provides an integrated human-centric and legal-methodological engineering framework, as well as a highly integrated federated learning and trusted industrial AI delivery framework, along with an executable cognitive Digital Twin system. The next steps of RE4DY initiative are focused on implementing and deploying an Open Toolkit and an open system based on DaaP containers. The main path that RE4DY will follow is to fully capitalize on the potential of the Edge-Cloud Continuum and overcome the complexities of data exchange. It involves offering a comprehensive suite of autonomous data engineering tools that seamlessly integrate sovereign Data Spaces and digital thread fabrics. Furthermore, it will provide compliance solutions that facilitate end-to-end and lifecycle digital thread operations within the context of industrial agreements. With a specific emphasis on resilience, RE4DY aims to establish a dynamic marketplace for data and applications, empowering agile manufacturing, and supply networks. Additionally, it will establish data pipelines and ensure continuous delivery of AI, enabling smooth data flow and driving efficient AI-powered processes.

Finally, when considering the impact of the proposed solution beyond the RE4DY project, the resources of the Edge-Cloud Continuum environment are set to redefine data management, facilitating agile processes and robust industrial value networks. The introduction of DaaP as an innovative concept, alongside frameworks like Data Fabric and Data Mesh, implies a considerable step towards modern digital data exchange solutions. These approaches not only empower data-driven decision-making but also provide a concrete base for managing and leveraging data in diverse commercial and business sectors. Furthermore, it effectively addresses the imperative of data sovereignty preservation, especially in alignment with European Data Initiatives.

References

1. Nagel, L., Lycklama, D.: Design Principles for Data Spaces – Position Paper (2021). https://doi.org/10.5281/ZENODO.5105744
2. Otto, B., Jarke, M.: Designing a multi-sided data platform: findings from the international data spaces case. Electron. Mark. **29**, 561–580 (2019). https://doi.org/10.1007/s12525-019-00362-x [Crossref]
3. Alliance DSB: Technical Convergence. https://data-spaces-business-alliance.eu/wp-content/uploads/dlm_uploads/Data-Spaces-Business-Alliance-Technical-Convergence-V2.pdf. Accessed 22 June 2023 (2023)
4. Huang G, He J, Chi CH, Zhou W, Zhang Y (2015) A data as a product model for future consumption of big stream data in clouds. Proc. 2015 IEEE Int. Conf. Serv. Comput. SCC 2015 256–263. https://doi.org/10.1109/SCC.2015.43
5. Data is a Product. https://www.ama.org/marketing-news/data-is-a-product/?utm_source=rss & utm_medium=rss&utm_campaign=data-is-a-product. Accessed 28 Sept 2023
6. Savaglio, C., Fortino, G.: A simulation-driven methodology for IoT data mining based on edge computing. ACM Trans. Internet Technol. **21** (2021). https://doi.org/10.1145/3402444
7. S-Julián, R., Lacalle, I., Vaño, R., Boronat, F., Palau, C.E.: Self-* capabilities of cloud-edge nodes: a research review. Sensors. **23**(23), 2931 (2023). https://doi.org/10.3390/S23062931 [Crossref]
8. Schulz, P., Matthe, M., Klessig, H., Simsek, M., Fettweis, G., Ansari, J., Ashraf, S.A., Almeroth, B., Voigt, J., Riedel, I., Puschmann, A., Mitschele-Thiel, A., Muller, M., Elste, T., Windisch, M.: Latency critical IoT applications in 5G: perspective on the design of radio interface and network architecture. IEEE Commun. Mag. **55**, 70–78 (2017). https://doi.org/10.1109/MCOM.2017.1600435CM [Crossref]
9. Byers, C.C.: Architectural imperatives for fog computing: use cases, requirements, and architectural techniques for fog-enabled IoT networks. IEEE Commun. Mag. **55**, 14–20 (2017). https://doi.org/10.1109/MCOM.2017.1600885 [Crossref]
10. Pingdom Webpages Are Getting Larger Every Year. https://www.pingdom.com/blog/webpages-are-getting-larger-every-year-and-heres-why-it-matters/. Accessed 20 June 2023
11. Industry IoT Consortium Edge computing. https://www.iiconsortium.org/wc-edge-computing/. Accessed 20 June 2023
12. Mostafavi, S.A., Dawlatnazar, M.A., Paydar, F.: Edge computing for IoT: challenges and solutions. J. Commun. Technol. Electron. Comput. Sci. **25**, 5–8 (2019). https://doi.org/10.22385/JCTECS.V25I0.293 [Crossref]

13. Satyanarayanan, M.: The emergence of edge computing. Computer (Long Beach Calif). **50**, 30–39 (2017). https://doi.org/10.1109/MC.2017.9
[Crossref]

14. Milojicic, D.: The edge-to-cloud continuum. Computer (Long Beach Calif). **53**, 16–25 (2020). https://doi.org/10.1109/MC.2020.3007297
[Crossref]

15. Rosendo, D., Costan, A., Valduriez, P., Antoniu, G.: Distributed intelligence on the edge-to-cloud continuum: a systematic literature review. J. Parallel Distrib. Comput. **166**, 71–94 (2022). https://doi.org/10.1016/J.JPDC.2022.04.004
[Crossref]

16. Pagano, P., Candela, L., Castelli, D.: Data interoperability. Data Sci. J. **12** (2013). https://doi.org/10.2481/DSJ.GRDI-004

17. Ganzha, M., Paprzycki, M., Pawlowski, W., Szmeja, P., Wasielewska, K., Palau, C.E.: From implicit semantics towards ontologies – practical considerations from the INTER-IoT perspective. 2017 14th IEEE Annu. Consum. Commun. Netw. Conf. CCNC 2017, 59–64 (2017). https://doi.org/10.1109/CCNC.2017.7983082

18. Satyanarayana Reddy, G., Srinivasu, R., Poorna, M., Rao, C., Rikkula, S.R.: Data warehousing, data mining, Olap and Oltp technologies are essential elements to support decision-making process in industries. Int. J. Comput. Sci. Eng. **02**, 2865–2873 (2010)

19. SchahczenskiCelia: Operational versus analytic data. J. Comput. Sci. Coll. (2017). https://doi.org/10.5555/3144605.3144637

20. Vaisman, A., Zimányi, E.: Data Warehouse Systems. Springer (2014)
[Crossref]

21. Mathis, C.: Data lakes. Datenbank-Spektrum. **17**, 289–293 (2017). https://doi.org/10.1007/s13222-017-0272-7
[Crossref]

22. Trakadas, P., Nomikos, N., Michailidis, E.T., Zahariadis, T., Facca, F.M., Breitgand, D., Rizou, S., Masip, X., Gkonis, P.: Hybrid clouds for data-intensive, 5G-enabled IoT applications: an overview, key issues and relevant architecture. Sensors. **19**, 3591 (2019). https://doi.org/10.3390/S19163591
[Crossref]

23. Lin, K., Gao, J., Li, Y., Savaglio, C., Fortino, G.: Multi-granularity collaborative decision with cognitive networking in intelligent transportation systems. IEEE Trans. Intell. Transp. Syst. **24**, 1088–1098 (2023). https://doi.org/10.1109/TITS.2022.3151754
[Crossref]

24. Al-Zahrani, F.A.: Subscription-based data-sharing model using Blockchain and data as a service. IEEE Access. **8**, 115966–115981 (2020). https://doi.org/10.1109/ACCESS.2020.3002823
[Crossref]

25. Boonma, P., Natwichai, J., Khwanngern, K., Nantawad, P.: Dahs: a distributed data-as-a-service framework for data analytics in healthcare. Lect. Notes. Data Eng. Commun. Technol. **24**, 486–495 (2019). https://doi.org/10.1007/978-3-030-02607-3_45/TABLES/1
[Crossref]

26. Data Spaces Support Centre. DSSC Glossary. https://dssc.eu/wp-content/uploads/2023/03/DSSC-Data-Spaces-Glossary-v1.0.pdf. Accessed 22 June 2023 (2023)

27. IDSA Functional Requirements – IDSA Rulebook V2. https://docs.internationaldataspaces.org/idsa-rulebook-v2/idsa-rulebook/3_functional_requirements. Accessed 26 June 2023

28. Saturi, R.: Data as a service (Daas) in cloud computing [data-as-a-service in the age of data] data as a service Daas in cloud computing. Glob. J. Comput. Sci. Technol. Cloud Distrib. **12** (2012)

29. TRUSTS Project: D2.1 ‘Definition and analysis of the EU and worldwide data market trends and industrial needs for growth.’ https://www.trusts-data.eu/wp-content/uploads/2021/07/D2.1-Definition-and-analysis-of-the-EU-and-worldwide-data-market-trends-....pdf. Accessed 22 June 2023 (2021)

30. Ghiran, A.M., Buchmann, R.A.: The model-driven enterprise data fabric: a proposal based on Conceptual modelling and knowledge graphs. Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). **11775 LNAI**, 572–583 (2019). https://doi.org/10.1007/978-3-030-29551-6_51/FIGURES/3
[Crossref]

31. Using Data Fabric Architecture to Modernize Data Integration. https://www.gartner.com/smarterwithgartner/data-fabric-architecture-is-key-to-modernizing-data-management-and-integration. Accessed 22 June 2023

32. Zaidi, E., Thoo, E., De Simoni, G., Beyer, M.: Data Fabrics Add Augmented Intelligence to Modernize Your Data Integration With Eric Thoo Gart Grou 17 (2019)

33. Hechler, E., Weihrauch, M., Wu, Y. (Catherine): Data fabric architecture patterns. Data Fabr Data Mesh Approaches with AI, 231–255 (2023). https://doi.org/10.1007/978-1-4842-9253-2_10

34. Yang, J., Tang, Y., Beheshti, A.: Design methodology for service-based data product sharing and trading. Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). **12521 LNCS**, 221–235 (2021). https://doi.org/10.1007/978-3-030-73203-5_17/FIGURES/4
[Crossref]

35. Dehghani, Z.: How to Move Beyond a Monolithic Data Lake to a Distributed Data Mesh, pp. 1–20 (2019)

36. Dehghani, Z. Data mesh principles and logical architecture. In: MartinFowler.com. https://martinfowler.com/articles/data-mesh-principles.html. Accessed 1 June 2023 (2020)

37. Machado, I.A., Costa, C., Santos, M.Y.: Data mesh: concepts and principles of a paradigm shift in data architectures. Procedia Comput. Sci. **196**, 263–271 (2022). https://doi.org/10.1016/J.PROCS.2021.12.013
[Crossref]

38. Goedegebuure, A., Kumara, I., Driessen, S., Di Nucci, D., Monsieur, G., Heuvel, W., van den Tamburri, D.A.: Data Mesh: A Systematic Gray Literature Review (2023)

39. Butte, V.K., Butte, S.: Enterprise data strategy: a decentralized data mesh approach. Int. Conf. Data Anal. Bus. Ind. ICDABI. **2022**, 62–66 (2022). https://doi.org/10.1109/ICDABI56818.2022.10041672
[Crossref]

40. Panigrahy, S, Dash, B., Thatikonda, R.: From Data Mess to Data Mesh: Solution for Futuristic Self-Serve Platforms (2023)

41. Bode, J., Kühl, N., Kreuzberger, D., Hirschl, S., Holtmann, C.: Data Mesh: Motivational Factors, Challenges, and Best Practices (2023)

42. Hechler, E., Weihrauch, M., Wu, Y. (Catherine): Key data fabric and data mesh capabilities. Data Fabr. Data Mesh Approaches with AI, 89–121 (2023). https://doi.org/10.1007/978-1-4842-9253-2_5

Footnotes

1 https://internationaldataspaces.org/

2 https://gaia-x.eu/

3 https://www.volkswagenautoeuropa.pt/

4 https://www.avioaero.com/en

5 https://bicindiana.com/

6 https://www.sipbb.ch/en/forschung/swiss-smart-factory/

OceanofPDF.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
S. Pal et al. (eds.), *IoT Edge Intelligence*, Internet of Things: Technology, Communications and Computing
https://doi.org/10.1007/978-3-031-58388-9_8

Security, Privacy, Trust, and Provenance Issues in Internet of Things–Based Edge Environment

Anand Kumar Mishra[1], Shrikant Tiwari[2], Amit Kumar Tyagi[3] ✉ and Micheal Olaolu Arowolo[4]
(1) Computer Science and Engineering, NIIT University, Neemrana, Rajasthan, India
(2) Department of Computer Science & Engineering, School of Computing Science and Engineering (SCSE), Galgotias University, Greater Noida, Uttar Pradesh, India
(3) Department of Fashion Technology, National Institute of Fashion Technology, New Delhi, India
(4) Department of Electrical Engineering and Computer Science, Bond Life Sciences Center, University of Missouri, Columbia, MO, USA

Keywords Security – Privacy – Trust – Smart environment – Edge computing

1 Introduction

The proliferation of Internet of Things (IoT) devices and the adoption of edge computing have made way for innovative and efficient solutions in various domains. However, the integration of IoT devices within an edge environment raises important issues regarding security, privacy, trust, and provenance. These issues are important as they directly impact the integrity, confidentiality, and reliability of data and services in IoT-based edge environments. Now each issue will be discussed:

- Security: In an IoT-based edge environment, security threats can arise from various sources, including malicious actors, vulnerabilities in IoT devices, insecure communication channels, and unauthorized access to sensitive data. Attacks such as device tampering, data breaches, denial-of-service, and malware injection raise important risks. Securing IoT devices, data, and infrastructure within the edge environment necessitates the implementation of strong security measures, encompassing authentication, access control, encryption, and intrusion detection.
- Privacy: Privacy issues arise in edge environments due to the substantial collection of personal and sensitive data by IoT/smart devices. The proximity of edge devices to end-users, coupled with their capacity to locally process data, increases the potential for unauthorized access, profiling, and surveillance. It is important to address these risks by implementing privacy-preserving techniques like data anonymization, differential privacy, and secure data sharing protocols. These measures are essential not only to safeguard individuals’ privacy rights but also to ensure compliance with data protection regulations.
- Trust: Building trust in IoT-based edge environments is important for users and stakeholders. Trust involves assessing the reliability, integrity, and authenticity of IoT devices, data, and services within the edge environment. Ensuring the trustworthiness of devices by verifying their identity, integrity, and compliance with security standards is essential. Moreover, establishing trusted communication channels, secure data exchange, and robust mechanisms for monitoring and verifying the behavior of IoT devices fosters trust among users and stakeholders.
- Provenance: Provenance refers to the ability to trace the origin, ownership, and history of data and assets in an IoT-based edge environment. Provenance plays a critical role in ensuring data integrity, enabling accountability, and detecting and mitigating data tampering or malicious activities. Implementing mechanisms to capture and preserve data provenance information, including metadata, timestamps, and digital signatures facilitates traceability, accountability, and auditability in the edge environment.

Hence, these security, privacy, trust, and provenance issues require a multi-faceted approach. It involves implementing robust security mechanisms, adopting privacy-by-design principles, establishing trust frameworks, promoting transparency, and integrating provenance tracking mechanisms within the IoT-based edge environment. Collaboration between industry, researchers, policymakers, and end-users is necessary to develop standards, best practices, and regulatory frameworks that address these issues effectively. By addressing these issues, IoT-based edge environments can provide secure, privacy-aware, trustworthy, and accountable solutions that unlock the full potential of the IoT and edge computing paradigm.

---

2 Related Work

Security and privacy have gained essential attention by several researcher in recent years. Several studies have focused on addressing these challenges to ensure the secure and trustworthy operation of IoT systems in edge computing environments. Here are some related works in this area.

In [1], the authors have provided a survey paper that provides an overview of security and privacy challenges in edge computing, including IoT-based edge environments. It discusses various threats, vulnerabilities, and attack scenarios, along with existing solutions and open research issues. Further, in [2], the authors explore privacy and security challenges in the IoT domain, covering different aspects of IoT, including edge environments. They discuss various security and privacy threats, existing solutions, and future research directions. Then, in [3], the authors presented a comprehensive perspective on trustworthy edge computing, addressing security, privacy, trust, and provenance issues. They discuss the importance of trustworthiness in edge environments and present a framework for ensuring trust and security in IoT-based edge computing systems. In [4], the authors explore the use of blockchain technology to enhance security, privacy, and trust in IoT systems, including edge environments. They discuss various blockchain-based solutions and their potential applications in addressing security and trust challenges in IoT. In [5], the authors focus on provenance management in IoT systems, including edge environments. They discuss the importance of provenance for ensuring data integrity, accountability, and trustworthiness in IoT-based edge computing. The paper presents various approaches, techniques, and challenges related to provenance in IoT.

These works provide a good starting point for understanding and exploring the research landscape related to security, privacy, trust, and provenance issues in IoT-based edge environments. Further research can be conducted based on these foundations to address specific challenges and develop novel solutions in this domain.

---

3 IoT-Based Edge Environment Applications

IoT-based edge environments provide a wide range of applications across various domains [6], using the capabilities of edge computing and the connectivity of IoT devices. Here are some examples of IoT-based edge environment applications:

- Smart Cities: The foundation of smart cities heavily depends on IoT-enabled edge environments. Edge devices, such as sensors and actuators, assume an important role in supervising and managing various facets of urban infrastructure. This encompasses monitoring and controlling elements such as traffic flow, parking, waste management, environmental conditions, and energy usage. Leveraging edge computing for real-time data analysis and decision-making enhances cities’ efficiency, sustainability, and overall livability.
- Industrial Automation: Industrial IoT (IIoT) applications benefit from edge environments by enabling real-time data analysis and localized decision-making within industrial settings. Edge devices can collect data from machines, sensors, and production lines, allowing for predictive maintenance, quality control, supply chain optimization, and remote monitoring. Edge computing reduces latency, enhances operational efficiency, and supports mission-critical industrial processes.
- Health-Care Monitoring: Internet of Things (IoT)-powered edge environments are instrumental in facilitating remote health-care monitoring and personalized health-care solutions. Devices situated at the edge, including wearables and medical sensors, possess the capability to continually monitor important signs, collect health data, and perform real-time analysis. This capability enables early detection of health issues, remote patient monitoring, and timely intervention, thereby enhancing patient care, reducing health-care costs, and improving the overall health-care experience.
- Agriculture and Farming: The integration of edge computing in agriculture facilitates precision farming and the implementation of smart agricultural practices. In on-field deployments, IoT devices can collect data on parameters like soil moisture, temperature, humidity, and crop

health. The edge environment processes this data locally, offering farmers real-time insights into aspects such as irrigation scheduling, pest control, and optimal fertilization. This approach enhances crop yield, resource efficiency, and overall sustainability in agriculture.

- Retail and Customer Experience: IoT-driven edge environments facilitate tailored customer experiences, effective inventory management, and precision-targeted marketing within the retail sector. Edge devices can track customer behavior, analyze purchase patterns, and provide real-time recommendations. Additionally, edge computing supports in-store operations, such as shelf monitoring, inventory tracking, and automated checkout systems, enhancing efficiency and customer satisfaction.
- Smart Homes and Building Automation: IoT-based edge environments play a role in advancing smart home automation and building management systems. Devices at the edge, like smart thermostats, lighting systems, and security cameras, empower remote control, energy optimization, and improved security. Edge computing facilitates localized decision-making, diminishing reliance on cloud services and ensuring quicker response times.
- Transportation and Logistics: IoT-based edge environments optimize transportation and logistics operations by monitoring vehicle conditions, tracking assets, and analyzing real-time traffic data. Edge devices installed in vehicles and logistics hubs can collect data on fuel consumption, route optimization, vehicle maintenance, and cargo tracking. This enables efficient logistics management, improved fleet operations, and enhanced supply chain visibility.

Note that these examples showcase the diverse applications that IoT-based edge environments can support in the future. With ongoing technological advancements and the continuous evolution of edge devices and networks, we anticipate the emergence of even more innovative applications and use cases across various industries and domains.

4 Available Simulation Tools to Implement Internet of Things-Based Edge Environment

There are various simulation tools accessible for implementing and simulating IoT edge environments. These tools provide a virtual environment to design, test, and analyze IoT systems, including edge computing components. Here are some popular simulation tools for implementing IoT edge environments:

- OMNeT++: This simulator is a discrete event simulation framework widely used for simulating communication networks and distributed systems, including IoT and edge environments. It provides a modular and extensible architecture that allows users to model and simulate complex systems. OMNeT++ provides various libraries and models specifically designed for simulating IoT and edge computing scenarios.
- Contiki: Contiki is an open-source operating system specifically designed for resource-constrained IoT devices. It provides a simulation environment called Cooja, which allows users to simulate IoT networks and edge environments. This simulator supports various IoT protocols and can simulate large-scale networks with realistic radio models. It is particularly useful for evaluating the performance of IoT applications and protocols.
- NetSim: NetSim is a commercial network simulator that provides comprehensive support for simulating IoT networks and edge environments. It provides a user-friendly interface and a wide range of IoT-specific models and protocols. This simulator allows users to design and simulate IoT applications, evaluate network performance, and analyze system behavior in edge computing scenarios.
- SimPy: SimPy stands out as a discrete-event simulation library designed for Python, serving as a valuable tool in constructing IoT edge environments. Boasting a straightforward yet potent framework, SimPy excels in modeling and simulating intricate systems. Users can leverage this simulator to craft simulation models encompassing IoT networks, edge devices, and communication protocols. SimPy offers flexibility for implementing bespoke simulation scenarios renowned for its high degree of customization.
- MATLAB/Simulink: MATLAB/Simulink is a widely used tool for modeling and simulation in various domains, including IoT and edge computing. It provides a comprehensive set of tools and libraries for designing and simulating IoT systems. This simulator provides a graphical environment for modeling and simulating IoT edge environments, allowing users to design and analyze system behavior.

- IoTIFY: IoTIFY is a cloud-based simulation platform specifically designed for simulating IoT environments. It allows users to create virtual IoT devices, design IoT networks, and simulate edge computing scenarios. This simulator supports a wide range of IoT protocols and provides real-time monitoring and visualization of simulated IoT systems.

Hence, these simulation tools provide a means to evaluate and optimize IoT edge environments before deployment, allowing users to assess system performance, analyze network behavior, and test different configurations and scenarios. The choice of simulation tool depends on specific requirements, level of complexity, and the desired features for the IoT edge environment being simulated.

---

5 Open Issues in Internet of Things-Based Edge Environment

While there are considerable potentials demonstrated by IoT-based edge environments, there are also numerous unresolved issues that must be addressed for their successful implementation and widespread adoption globally. Some of the key open issues in IoT-based edge environments are as follows:

- Security and Privacy: Security and privacy continue to face important challenges in IoT-based edge environments [7]. Given the frequently constrained resources of edge devices and potential deficiencies in security mechanisms, they become susceptible to various attacks. Additionally, the proximity of edge devices to end-users raises issues regarding the privacy aspects of collecting, storing, and processing personal data. It is imperative to counter these challenges by establishing robust security measures, incorporating encryption techniques, and implementing privacy-preserving mechanisms.
- Interoperability and Standards: Interoperability poses a challenge in IoT devices and edge computing systems, as they are frequently crafted by diverse manufacturers and may operate on distinct communication protocols. The absence of standardized practices impedes the smooth integration and communication between various devices and platforms. Establishing industry-wide interoperability standards and protocols is

imperative to guarantee compatibility, foster collaboration, and enhance scalability within IoT-based edge environments.

- Resource Constraints: Edge devices typically have limited computational power, memory, and energy resources. This raises challenges in implementing resource-intensive AI algorithms, performing complex analytics, and managing data storage and processing. Optimizing algorithms for edge computing, implementing energy-efficient mechanisms, and exploring hardware acceleration techniques can help mitigate these resource constraints.
- Data Management and Analytics: IoT generates substantial data, facing important challenges for effective management and analysis in edge environments. The limited storage capacity of edge devices and constraints on transmitting all data to the cloud, given bandwidth limitations and latency requirements, need necessary strategic data management. Developing efficient strategies such as data filtering, aggregation, and intelligent data analytics at the edge can facilitate real-time information processing and alleviate the burden on cloud resources.
- Scalability and Management: Managing a large number of heterogeneous edge devices distributed across different locations requires robust management frameworks. Device discovery, configuration, software updates, and monitoring become complex tasks in edge environments. Developing scalable management systems, incorporating automation, and using artificial intelligence for device management can streamline operations and ensure efficient scaling.
- Energy Efficiency: Energy consumption is a critical issue in IoT-based edge environments, especially for battery-powered devices. Edge devices need to operate with minimal energy consumption to prolong battery life and reduce environmental impact. Optimizing power management, implementing energy harvesting techniques, and exploring energy-efficient communication protocols are key areas of focus to enhance energy efficiency in edge environments.
- Edge-Cloud Integration: While edge computing provides localized data processing and real-time analytics, there is still a need for seamless integration and collaboration between edge devices and cloud resources [8, 9]. Offloading computational tasks to the cloud can enable complex analytics, resource sharing, and centralized management. Developing efficient edge-cloud integration mechanisms and load balancing

strategies is essential for maximizing the potential of both edge and cloud computing.

Note that these open issues require collaborative efforts from researchers, industry experts, policymakers, and standardization bodies. By overcoming these challenges, IoT-based edge environments can unlock their full potential and revolutionize various domains, ranging from smart cities and industrial automation to health care and agriculture.

5.1 Security Issues in IoT-Based Edge Environment

Security poses an important issue in IoT-based edge environments owing to the distributed nature of devices and the potential vulnerabilities they might have. Here are some key security issues that need to be addressed in IoT edge environments:

- Device Security: IoT devices deployed in edge environments may have limited resources and may not have robust security mechanisms in place. This makes them susceptible to various attacks, such as device tampering, unauthorized access, and firmware manipulation. Ensuring secure device authentication, strong access control mechanisms, and regular firmware updates are essential to mitigate device-level security risks.
- Communication Security: IoT devices in edge environments communicate with each other and with the cloud, creating multiple points of vulnerability. Insecure communication channels can expose sensitive data to interception, eavesdropping, and tampering. Deploying secure communication protocols, such as Transport Layer Security (TLS) or Datagram Transport Layer Security (DTLS), and encrypting data during transit can effectively guard against unauthorized access and prevent data breaches.
- Data Security: IoT devices generate and transmit substantial volumes of data, often containing sensitive information. Ensuring the secure storage and processing of this data presents challenges in maintaining confidentiality and integrity. Utilizing data encryption, establishing access control mechanisms, and following secure data storage practices are important measures to prevent unauthorized data access and mitigate the risk of data leakage.
- Edge-to-Cloud Security: Edge environments often rely on cloud resources for tasks such as data analytics and storage. Securing the

communication and data exchange between edge devices and the cloud is important. Techniques such as secure authentication, data encryption, and secure APIs can ensure the confidentiality and integrity of data as it moves between the edge and the cloud.
- Edge Device Management: Supervising a substantial number of edge devices dispersed across various locations presents security challenges. Unauthorized access to edge devices or compromised management interfaces can result in essential security breaches. Implementing secure device management practices, such as robust authentication, access control, and routine security updates, is important to thwart unauthorized device access and uphold device integrity.
- Physical Security: Edge devices deployed in various locations are vulnerable to physical attacks, theft, and tampering. Securing the physical environment where edge devices are deployed, implementing tamper-resistant measures, and monitoring for unauthorized physical access can help protect against physical security threats.
- Security Monitoring and Incident Response: Regularly monitoring edge devices, network traffic, and system logs is important for promptly detecting and responding to security incidents. Implementing intrusion detection systems, security information and event management (SIEM) solutions, and incident response plans is important in efficiently identifying and mitigating security breaches.

Hence, addressing these security issues in IoT-based edge environments requires a multi-layered and holistic approach. It involves implementing security best practices at the device, communication, data, and management levels. Collaboration between device manufacturers, software developers, network providers, and security experts is important to ensure comprehensive security measures in IoT edge environments and safeguard against potential threats.

5.2 Privacy Issues in IoT-Based Edge Environment

Privacy is an essential issue in IoT-based edge environments due to the collection, processing, and storage of personal and sensitive data [10]. Here are some key privacy issues that need to be addressed in IoT edge environments:

- Data Collection and Consent: In edge environments, IoT devices frequently gather personal and sensitive data from individuals. Ensuring proper consent and transparency in data collection practices is important. Users should be provided with clear information about the data collected, its intended use, and the option to provide or withdraw consent.

- Data Minimization and Purpose Limitation: Privacy issues may arise as IoT devices tend to amass more data than necessary for their designated purpose. Mitigating these risks involves embracing data minimization practices, where only essential data is collected, and data retention is limited to the required duration. Well-defined policies should be established, outlining the purposes for collecting and utilizing the data.
- Data Security: Ensuring the security of data in IoT edge environments is important for safeguarding privacy. Implementing robust encryption, access controls, and secure storage mechanisms acts as a preventive measure against unauthorized access, data breaches, and leaks that could compromise individuals' privacy.
- User Control and Transparency: Providing users with control over their data and empowering them to make decisions about its use are of utmost importance. Offering choices for users to oversee their data, encompassing access, rectification, deletion, or restriction of data processing, enhances privacy. Transparency in data practices, including providing clear privacy policies and data processing notices, facilitates informed decision-making by users.
- Data Sharing and Third-Party Access: In IoT-based edge environments, there may be data sharing with third parties, such as cloud service providers or analytics platforms. Ensuring that data sharing agreements and contracts incorporate privacy safeguards, adhere to data protection standards, and impose restrictions on data usage by third parties is important to protect individuals' privacy rights.
- Anonymization and Pseudonymization: To protect privacy, IoT data should undergo processing that minimizes the identifiability of individuals. Anonymization techniques, such as the removal or encryption of personally identifiable information, can aid in protecting privacy while enabling data analysis and utilization.
- Privacy by Design: Incorporating privacy considerations from the outset of IoT edge environment development is important. Adhering to Privacy by Design principles entails integrating privacy safeguards into the design and architecture of IoT systems, deploying privacy-enhancing technologies, and conducting privacy impact assessments.

- Regulatory Compliance: Compliance with privacy regulations and data protection laws, such as the General Data Protection Regulation (GDPR), is essential in IoT edge environments. Organizations must understand and adhere to the relevant legal requirements, including obtaining necessary consents, providing individuals with privacy rights, and implementing appropriate data protection measures.

Hence, addressing privacy issues in IoT-based edge environments requires a comprehensive strategy that combines technical, organizational, and legal measures. Collaborative efforts involving IoT device manufacturers, developers, policymakers, and privacy experts are important to embed privacy considerations throughout the design, implementation, and operation of IoT edge environments. This collaborative approach ensures the protection of individuals’ privacy while harnessing the advantages of IoT technology.

5.3 Trust Issues in IoT-Based Edge Environment

Trust is an important factor in IoT-based edge environments, as it involves the interaction of multiple entities and the reliance on data and services provided by these entities. Here are some key trust issues that need to be addressed in IoT-based edge environments:

- Device Trustworthiness: Trustworthiness of IoT devices deployed in edge environments is important. It involves ensuring that the devices are authentic, reliable, and free from tampering or compromise. Establishing mechanisms for device authentication, integrity verification, and secure bootstrapping is essential to build trust in the devices and prevent malicious actors from exploiting vulnerabilities.
- Data Integrity and Authenticity: Ensuring the trustworthiness of data collected and processed in IoT edge environments requires safeguarding its integrity and authenticity. In the absence of proper measures, data may be subject to manipulation or tampering, resulting in inaccurate information and decision-making. The implementation of data integrity mechanisms, such as digital signatures and cryptographic techniques, proves instrumental in verifying the integrity and authenticity of data.
- Service Provider Trust: Trust in service providers, including cloud providers, platform providers, and third-party service providers, is important in IoT edge environments. Users rely on these providers for data storage, analytics, and other services. Assessing the reputation,

reliability, and security practices of service providers, along with strong service level agreements (SLAs) and contractual agreements, helps establish trust and ensures that services are delivered as promised.

- Trust in Data Sharing: In IoT edge environments, data sharing commonly occurs among various entities, including devices, applications, and platforms. Building trust in data sharing necessitates mechanisms to enforce data access controls, ensure data privacy, and maintain accountability for data handling. Implementation of secure data sharing protocols, data usage agreements, and audit trails can contribute to establishing trust in data sharing processes.
- Trust in System Resilience: IoT edge environments should be resilient to failures, attacks, and disruptions. Ensuring the availability, reliability, and resilience of the system builds trust among users and stakeholders. Implementing redundancy, fault tolerance, disaster recovery mechanisms, and conducting regular system audits and assessments contribute to the trustworthiness of the IoT edge environment.
- Trust in Privacy and Consent Management: Privacy and consent management mechanisms are essential in building trust in IoT edge environments [11, 12]. Users should have control over their personal data, understand how their data is being used, and be able to provide or withdraw consent. Implementing transparent privacy practices, user-friendly consent management interfaces, and adherence to privacy regulations fosters trust among users.
- Trust in Security Practices: Establishing trust in IoT edge environments requires implementing robust security practices. This includes deploying strong authentication, encryption, access controls, and security monitoring. Conducting regular security audits, vulnerability assessments, and adhering to security standards and best practices contribute to instilling trust in the system's security.

Therefore, addressing trust issues in IoT-based edge environments necessitates a combination of technical, organizational, and policy measures. Collaboration among stakeholders, including device manufacturers, service providers, regulators, and users, is essential to establish and maintain trust in IoT edge environments. Transparency, accountability, and adherence to industry standards and best practices are key elements in building trust and ensuring the successful adoption and acceptance of IoT-based edge environments.

5.4 Provenance Issues in IoT-Based Edge Environment

As in [13–15], provenance refers to the origin, history, and lineage of data, devices, and software components in IoT edge environments. Provenance issues in IoT-based edge environments can impact data reliability, trustworthiness, and accountability. Here are some key provenance issues that need to be addressed:

- Data Source Validation: In IoT-based edge environments, data is collected from various sources, including sensors, devices, and external systems. Validating the authenticity and integrity of these data sources is important to ensure their trustworthiness. Provenance information, which includes device identity, data collection timestamps, and sensor calibration details, plays an important role in establishing the reliability of these data sources.
- Data Transformation Tracking: Data in IoT edge environments may undergo various transformations, aggregations, and processing steps before being used for analysis or decision-making. Tracking and recording the transformations applied to data, along with the associated algorithms and parameters, is important to understand the lineage of the data and ensure its quality.
- Software Component Provenance: IoT-based edge environments involve the deployment of software components, such as firmware, operating systems, and analytics algorithms, on edge devices. Understanding the provenance of these software components, including their origin, version, and dependencies, is important to assess their trustworthiness and identify potential security vulnerabilities or compatibility issues.
- Device Lifecycle Tracking: IoT-enabled edge devices have a lifecycle that includes manufacturing, distribution, deployment, and decommissioning phases. Tracking the lifecycle of devices, including their ownership, maintenance history, and software updates, helps ensure the integrity and security of the devices throughout their lifespan.
- Trust in Device Configuration: Configuring edge devices with appropriate settings and security configurations is essential for their reliable and secure operation. Tracking the configuration changes made to devices, including the parameters and responsible entities, helps in identifying potential configuration errors or unauthorized modifications that could impact the trustworthiness of the device.

- Data Access and Usage Tracking: Provenance information should include details about who accessed the data, when it was accessed, and for what purpose. Tracking data access and usage helps in establishing accountability, ensuring compliance with data usage policies, and identifying potential privacy breaches or unauthorized data access.
- Chain of Custody: In some IoT-based edge environments, data may need to be shared or exchanged between multiple entities or systems. Establishing a clear chain of custody, including the entities involved, timestamps, and data integrity checks, ensures transparency and accountability in data handling and prevents unauthorized modifications or tampering.

Hence, provenance issues in IoT-based edge environments require the implementation of mechanisms for capturing, recording, and managing provenance information throughout the lifecycle of data, devices, and software components. This involves the use of technologies such as blockchain, distributed ledgers, and metadata management systems to track and authenticate provenance information. Collaborative efforts among stakeholders, including device manufacturers, data providers, software developers, and regulators, are important in defining standards and best practices for provenance management in IoT edge environments.

5.5 Scalability Issues in IoT-Based Edge Environment

Scalability is an essential challenge in IoT-based edge environments due to the large number of devices, data volume, and distributed nature of the system. Here are some key scalability issues that need to be addressed in IoT-based edge environments:

- Device Management: Managing a large number of IoT devices deployed in edge environments can be challenging. Scalability issues arise in terms of device provisioning, configuration management, software updates, and monitoring. Implementing efficient device management protocols, automated provisioning processes, and centralized management systems can help overcome these scalability challenges.
- Data Processing and Analytics: IoT edge environments produce a substantial amount of data requiring real-time or near-real-time processing and analysis. It is important to scale data processing and analytics capabilities to manage the growing data volume and complexity. Employing distributed computing technologies, including

edge computing frameworks, parallel processing, and load balancing techniques, can facilitate efficient and scalable data processing.
- Communication and Network Scalability: IoT edge environments depend on communication networks for transmitting data among devices, edge nodes, and cloud services. Scaling the network infrastructure to meet the growing number of devices, data traffic, and bandwidth requirements is important. This may entail optimizing network protocols, strategically deploying edge computing nodes, and leveraging technologies such as network function virtualization (NFV) or software-defined networking (SDN) to enhance network scalability.
- Resource Constraints: In edge environments, IoT devices commonly face constraints in computational power, memory, and energy resources. Scaling the system while mindful of these limitations poses a challenge. Overcoming scalability constraints imposed by resource-constrained devices can be achieved by optimizing resource usage through the adoption of lightweight protocols, leveraging edge intelligence, and employing distributed computing techniques.
- Security and Privacy: As the number of devices and data in IoT edge environments increases, scaling security and privacy measures becomes important. Ensuring secure device authentication, data encryption, access control mechanisms, and privacy safeguards at scale is a complex task. Employing scalable security architectures, threat detection and response systems, and privacy-enhancing technologies can address these scalability challenges.
- Interoperability and Standards: Interoperability challenges arise in IoT environments when devices from various manufacturers employ different communication protocols and data formats. To scale interoperability efforts, promoting standardization, fostering industry collaborations, and adopting open protocols and data formats can facilitate seamless integration and enhance scalability in IoT edge environments.
- System Monitoring and Management: Monitoring and managing the overall health, performance, and reliability of the IoT edge environment at scale can be challenging. Real-time monitoring, fault detection, and predictive maintenance techniques can help identify and resolve issues proactively, ensuring the scalability and availability of the system.

Hence, addressing scalability issues in IoT-based edge environments requires a combination of architectural design, technological advancements, and operational practices. It necessitates scalable hardware and software solutions, efficient communication networks, standardized interfaces, and automated management systems. Collaboration among stakeholders, including device manufacturers, network providers, software developers, and system integrators, is important to tackle scalability challenges and ensure the successful deployment and operation of IoT-based edge environments at scale.

5.6 Legal Issues in IoT-Based Edge Environment

Legal issues in IoT-based edge environments can arise due to the complexities and unique characteristics of these systems. Here are some key legal issues:

- Data privacy: IoT devices produce and accumulate extensive data, often comprising personal information. Regulatory frameworks like the General Data Protection Regulation (GDPR) in the European Union and the California Consumer Privacy Act (CCPA) in the United States govern the collection, storage, and processing of personal data. Organizations are required to comply with these regulations, ensuring adherence through measures such as obtaining explicit consent, implementing robust data protection measures, and maintaining transparency in data handling practices.
- Security breaches: IoT devices are susceptible to security breaches, posing risks of unauthorized access to sensitive information or potential physical harm. Organizations need to implement comprehensive security measures for their IoT systems, including encryption, access controls, and routine security audits. In the event of a breach, organizations may face legal liabilities, with potential obligations to notify affected individuals or regulatory authorities.
- Intellectual property rights: IoT devices often involve various intellectual property (IP) rights, such as patents, copyrights, and trademarks. Organizations must ensure that their IoT solutions do not infringe on the IP rights of others. Additionally, they need to protect their own IP by securing proper patents or copyrights for their innovative technologies.
- Product liability: IoT devices can have direct physical implications, such as in industrial settings or health-care environments. If a device malfunctions or causes harm, the manufacturer or service provider may be held liable for any damages or injuries. Compliance with relevant

product safety regulations, conducting risk assessments, and providing clear instructions and warnings are important to mitigate product liability risks.
- Regulatory compliance: IoT implementations may encounter diverse regulatory demands contingent on the industry and geographical context. Industries such as health care, energy, or transportation, for instance, may be subject to specific regulations and standards. It is imperative for organizations to comprehend and comply with relevant regulations, certifications, and standards to sidestep legal ramifications and guarantee the safety and reliability of their IoT systems.
- Contractual agreements: The implementation of IoT involves various stakeholders, encompassing manufacturers, service providers, vendors, and customers. Establishing transparent contractual agreements is important to delineate responsibilities, liabilities, data ownership, and arrangements for data sharing among these parties. Contracts should comprehensively address aspects such as data access, data protection, indemnification, and mechanisms for resolving disputes.
- Cross-border data transfer: The functioning of IoT systems often involves the cross-border transfer of data. Adhering to international data transfer regulations, particularly the limitations specified in the GDPR regarding the transfer of personal data to countries lacking sufficient data protection laws, is important. Organizations must comply with applicable data transfer mechanisms, such as standard contractual clauses or binding corporate rules, to ensure lawful and compliant data transfers.

Note that it is difficult for organizations and individuals involved in IoT-based edge environments to consult with legal professionals who specialize in data protection, intellectual property, cybersecurity, and regulatory compliance to navigate these legal issues effectively.

---

6 Critical Challenges Toward IoT-Based Edge Environment

Today IoT-based edge environments present several critical challenges that need to be addressed for successful implementation and operation [16]. Here are some of the key challenges:

- Scalability: Handling and expanding the infrastructure become intricate with the growing number of IoT devices and edge nodes. Edge environments typically encompass a distributed network of devices with diverse capabilities and connectivity. Guaranteeing smooth scalability and effective management of a substantial number of devices and nodes face important challenge.
- Connectivity and interoperability: Interoperability challenges arise in IoT devices and edge nodes due to the utilization of diverse communication protocols and technologies. Ensuring smooth connectivity and interoperability among devices from various manufacturers, platforms, or generations is important for facilitating efficient data exchange and collaboration within an edge environment.
- Edge computing capabilities: Edge environments strive to handle data in proximity to the source, diminishing latency and bandwidth demands. Nonetheless, edge devices frequently possess restricted computing resources and storage capacities compared to cloud-based systems. Enhancing the computational capabilities and storage capacity of edge devices while ensuring streamlined data processing and analytics remains an important challenge.
- Security and privacy: IoT devices and edge nodes can be vulnerable to security breaches and attacks. Edge environments introduce additional security challenges due to distributed networks, diverse devices, and potential physical access. Ensuring end-to-end security, including secure device onboarding, data encryption, access controls, and regular security updates, is important to mitigate risks and protect sensitive data.
- Data management and analytics: IoT devices generate massive volumes of data, and processing this data efficiently in an edge environment is a challenge. Edge nodes may have limited storage and computational capabilities, requiring careful data management strategies, such as data filtering, aggregation, and prioritization. Implementing efficient data analytics and machine-learning algorithms in resource-constrained edge environments is also challenging.
- Reliability and fault tolerance: Edge environments often involve distributed systems with numerous devices and nodes. Ensuring high reliability, fault tolerance, and resiliency is important for maintaining uninterrupted services. Redundancy, fault detection mechanisms, and

disaster recovery plans need to be implemented to minimize downtime and ensure system availability.

- Regulatory and legal issues: IoT-based edge environments may be subject to various legal and regulatory requirements, as discussed in the previous response. Complying with privacy regulations, data protection laws, and industry-specific standards can be challenging, particularly when operating in multiple jurisdictions.
- Energy efficiency: Many IoT devices in edge environments are battery-powered or have limited power sources. Maximizing energy efficiency and extending battery life are important for ensuring the sustainable operation of edge devices. Implementing strategies like optimizing power consumption, employing low-power communication protocols, and utilizing energy harvesting techniques is important to meet this challenge.
- System complexity and integration: Edge environments often involve the integration of diverse technologies, devices, platforms, and applications. Managing the complexity of integrating these components, ensuring seamless communication, and maintaining compatibility across the entire system is a major challenge. Standardization efforts and open interfaces can help alleviate integration challenges.
- Human factors: Finally, considering the human element in IoT-based edge environments is important. Training personnel, educating users, and creating user-friendly interfaces and experiences are important for effective adoption and utilization of edge systems. Addressing the skill gap and providing adequate training and support for managing edge systems can contribute to successful deployment and operation.

Hence, resolving the important challenges outlined above requires cooperation among various stakeholders, such as technology providers, manufacturers, policymakers, and end-users. Research and development efforts, commitment to industry standards, and ongoing innovation will be instrumental in overcoming these challenges and realizing the complete potential of IoT-based edge environments.

6.1 Technical Challenges in IoT-Based Edge Environment

In today’s era, IoT-based edge environments introduce several technical challenges that need to be addressed for successful implementation and operation [17]. Here are some of the key technical challenges:

- Connectivity and Communication: IoT devices in an edge environment need to establish reliable and seamless connectivity with each other and with the edge computing infrastructure. However, edge devices may operate in heterogeneous networking environments, including wired, wireless, and cellular networks, which can introduce connectivity challenges. Ensuring robust communication protocols, managing network congestion, and addressing intermittent connectivity are essential challenges.
- Edge Computing Resources: Edge devices typically have limited computing resources compared to cloud-based systems. Balancing the computational requirements of IoT applications with the constraints of edge devices is an important challenge. Optimizing resource utilization, implementing efficient data processing algorithms, and managing the trade-off between local processing and offloading to the cloud are key issues.
- Data Management and Storage: IoT devices produce extensive data, and effectively handling this data within an edge environment poses challenges. Edge devices frequently contend with restricted storage capacities, requiring the implementation of intelligent data management strategies. These strategies encompass techniques such as data filtering, aggregation, compression, and prioritization to reduce storage demands while retaining the most essential data for analysis.
- Data Security and Privacy: Securing data in an IoT-based edge environment is important due to the decentralized structure of the infrastructure. Edge devices are susceptible to physical attacks, potential data breaches, and unauthorized access. To protect sensitive data and ensure privacy, it is important to deploy robust security measures. These measures include encryption, access controls, secure bootstrapping, and the use of secure communication protocols.
- Edge-Cloud Integration: Integrating IoT systems seamlessly between edge and cloud computing components is a common requirement. Overcoming challenges such as efficient data transfer between the edge and the cloud, maintaining data consistency, and managing data synchronization and latency is important. Developing effective protocols, data caching strategies, and synchronization mechanisms is essential for achieving efficient and reliable edge-cloud integration.

- Device Management and Configuration: Managing a large number of IoT devices in an edge environment can be challenging. Device provisioning, configuration, monitoring, and firmware updates need to be handled efficiently. Over-the-air (OTA) update mechanisms, device management platforms, and automated configuration management are essential for managing and maintaining a diverse fleet of devices.
- Edge Analytics and Machine Learning: Performing data analytics and machine learning tasks at the edge introduces challenges due to limited computational resources. Optimizing algorithms for edge devices, implementing efficient edge analytics frameworks, and using techniques like federated learning to train models without centralizing data are important issues.
- Quality of Service and Service-Level Agreements: IoT applications often have specific quality of service (QoS) requirements, such as latency, reliability, and availability. Meeting these requirements in an edge environment, where network conditions and resource limitations vary, can be challenging. Designing QoS-aware systems, monitoring performance metrics, and establishing Service Level Agreements (SLAs) with clear expectations are important for ensuring a satisfactory user experience.
- Testing and Validation: Validating the functionality, reliability, and performance of IoT systems in an edge environment is complex. Testing edge devices, network connectivity, and interactions with cloud services require comprehensive test strategies. Additionally, creating realistic test environments, simulating edge conditions, and ensuring interoperability among devices and platforms are important challenges in the testing phase.
- Energy Efficiency: Energy efficiency stands as major issue in IoT-based edge environments, particularly for devices powered by batteries. Prioritizing power optimization, incorporating low-power communication protocols, and advancing energy harvesting techniques are important measures to extend battery life and ensure sustainable operation.

Note that these technical challenges require a multidisciplinary approach, involving expertise in networking, distributed computing, security, data management, and system integration. Ongoing research, standardization efforts, and collaboration among industry stakeholders are important for developing innovative solutions and best practices in IoT-based edge environments.

6.2 Non-technical Challenges in IoT-Based Edge Environment

In addition to technical challenges, IoT-based edge environments also face several non-technical challenges. These challenges are often related to organizational, societal, and regulatory aspects. Here are some key non-technical challenges in IoT-based edge environments:

- Business Model Transformation: Adopting IoT-based edge solutions may require organizations to undergo important business model transformations. This involves redefining roles, responsibilities, and revenue streams. Organizations need to evaluate the economic feasibility, value proposition, and long-term sustainability of their IoT-based edge initiatives. Identifying new business opportunities, developing partnerships, and adapting to new market dynamics are major challenges.
- Stakeholder Collaboration: IoT-based edge environments encompass a range of stakeholders, such as device manufacturers, service providers, infrastructure providers, regulators, and end-users. Collaborating and aligning interests among these stakeholders can be challenging, given their diverse objectives, priorities, and perspectives. Establishing effective governance structures, fostering collaboration, and addressing conflicting interests are important for the successful implementation and operation of such environments.
- Data Ownership and Governance: The considerable data generated by IoT devices prompts issues regarding data ownership, control, and governance. Deliberations on who owns the data, who holds access rights, and how it can be utilized demand careful attention. Organizations must institute data governance frameworks, delineate data-sharing agreements, and ensure transparency in data handling to instill trust and alleviate potential disputes.
- Regulatory and Compliance Issues: IoT-based edge environments fall under diverse legal and regulatory frameworks, encompassing data protection, privacy, cybersecurity, and industry-specific regulations. Navigating compliance with these regulations, which may vary across jurisdictions, poses challenges for organizations. Keeping abreast of evolving regulations, implementing requisite controls, and ensuring

transparency in data handling practices are essential to mitigate legal and compliance risks.

- User Acceptance and Adoption: The successful adoption of IoT-based edge solutions relies on end-user acceptance. However, there may be resistance or reluctance from end-users due to issues about privacy, security, usability, or job displacement. Educating and creating awareness among end-users, addressing their issues, and providing user-friendly interfaces and experiences are important for widespread adoption and utilization.
- Skill Gap and Workforce Training: Deploying and managing IoT-based edge environments requires a skilled workforce. However, there is often a skill gap in areas such as edge computing, cybersecurity, data analytics, and system integration. Organizations need to invest in workforce training and development programs to build the necessary expertise and ensure that employees can effectively operate and maintain IoT-based edge systems
- Sustainability and Environmental Impact: The deployment of IoT devices and edge infrastructure may have environmental repercussions, including heightened energy consumption and electronic waste. Striking a balance between the advantages of IoT-based edge solutions and environmental sustainability presents a challenge. Incorporating energy-efficient practices, accounting for the lifecycle impact of devices, and advocating responsible disposal and recycling practices are important for minimizing the environmental footprint.
- Cost and Return on Investment: Implementing IoT-based edge environments requires initial investments in infrastructure, devices, and system integration. Organizations must thoroughly assess the costs and anticipate the return on investment (ROI) to justify the implementation. The identification of cost-effective solutions, conducting comprehensive cost-benefit analyses, and demonstrating tangible benefits and ROI are important steps for securing the necessary funding and support.

In summary, high-profile security breaches and privacy incidents can erode public trust in IoT-based edge environments. Building trust among end-users and stakeholders by prioritizing security, implementing privacy-by-design principles, and being transparent about security practices are important for the next generation.

6.3 Research Challenges in IoT-Based Edge Environment

Research challenges in IoT-based edge environments encompass a wide range of topics across various disciplines. Here are some key research challenges:

- Edge Computing Architectures: Designing efficient and scalable edge computing architectures that can handle the increasing volume and diversity of IoT devices is important research challenge. This includes exploring novel architectural models, such as hierarchical, fog, or multi-tiered architectures, and developing algorithms for dynamic resource management, task scheduling, and load balancing in edge environments.
- Resource Optimization: Optimizing the allocation and utilization of computing, storage, and networking resources in edge environments is important to meet the diverse requirements of IoT applications. Research is needed to develop intelligent resource allocation algorithms, energy-efficient computation techniques, and adaptive resource management strategies to ensure optimal resource usage and performance.
- Data Management and Analytics: A key research challenge involves creating efficient and scalable techniques for data management and analytics within edge environments. This encompasses the exploration of mechanisms for data storage and retrieval, techniques for data preprocessing and filtering, algorithms for distributed data analytics, and approaches for machine learning and deep learning at the edge. These endeavors aim to enable real-time, low-latency data processing and the timely generation of valuable information.
- Security and Privacy: Enhancing the security and privacy of IoT-based edge environments represents an important focus in research. Tackling challenges associated with secure device onboarding, authentication, access control, secure communication protocols, and privacy-preserving data analysis techniques is imperative. Moreover, delving into advanced security mechanisms, including blockchain-based solutions, secure multi-party computation, and anomaly detection, is essential to effectively mitigate security risks.
- Edge-Cloud Integration and Cooperation: Investigating efficient mechanisms for seamless integration and collaboration between edge and cloud computing components is a research challenge. This includes exploring hybrid architectures, data synchronization and consistency

mechanisms, edge-cloud coordination protocols, and workload migration strategies to enable dynamic and efficient utilization of resources across the edge-cloud continuum.

- Communication and Networking: Developing robust and scalable communication and networking solutions for IoT-based edge environments is a research challenge. This includes exploring efficient communication protocols, adaptive routing mechanisms, congestion control algorithms, and quality of service (QoS) management techniques to ensure reliable and low-latency communication in heterogeneous edge networks.
- Trust, Reliability, and Resilience: Enhancing trust, reliability, and resilience in IoT-based edge environments is an important research area. This includes investigating fault tolerance mechanisms, self-healing techniques, edge-based anomaly detection, and intrusion detection systems to ensure system integrity and availability. Additionally, exploring reputation management systems, trust models, and accountability mechanisms is important for establishing trust in edge environments.
- Standards and Interoperability: A research challenge lies in the development of standards and protocols that guarantee interoperability and compatibility across a spectrum of IoT devices and edge platforms. Research endeavors should concentrate on defining shared data models, communication protocols, and interoperability frameworks. This focus is important to facilitate smooth integration and collaboration among different devices, platforms, and ecosystems.
- Edge Artificial Intelligence (AI) and Machine Learning: Exploring AI and machine-learning techniques tailored for edge environments is a research challenge. This includes developing lightweight and energy-efficient machine learning algorithms, edge-based model training and inference approaches, distributed learning frameworks, and federated learning techniques to enable intelligent decision-making and data analytics at the edge.
- Energy Efficiency and Sustainability: Investigating energy-efficient practices, energy harvesting techniques, and sustainable design principles for IoT-based edge environments is a research area of growing importance. Research should focus on optimizing energy consumption, developing energy-aware scheduling algorithms, exploring energy-

efficient communication protocols, and using renewable energy sources to ensure the sustainability of edge systems.

Hence, these research challenges require collaboration between academia, industry, etc. Researcher/scientist efforts should aim to develop innovative solutions, validate proposed approaches through real-world deployments and experimentation, and contribute to the development of standards and best practices in IoT-based edge environments.

---

7 Future Research Opportunities with IoT-Based Edge Environment

The field of IoT-based edge environments provides several exciting research opportunities for future exploration and advancement. Here are some potential research areas that hold promise:

- Edge Intelligence and Autonomy: Investigating how to enable intelligent decision-making and autonomy at the edge is an intriguing research opportunity. This includes developing advanced edge analytics algorithms, machine learning models, and AI techniques that can autonomously process and analyze data at the edge. Exploring context-aware and adaptive edge intelligence approaches can lead to more efficient and intelligent edge systems.
- Edge Computing for Augmented Reality/Virtual Reality (AR/VR): Exploring the integration of edge computing with AR/VR technologies is an emerging research area. Investigating how edge resources can enhance the performance and user experience of AR/VR applications by reducing latency, improving image rendering, and enabling real-time interactions opens up new possibilities for immersive and interactive edge-based AR/VR systems.
- Edge Computing for Internet of Medical Things (IoMT): The health-care sector can greatly benefit from IoT-based edge environments. Research can focus on developing edge-based solutions for real-time patient monitoring, remote health-care services, and smart health-care systems. Investigating data security, privacy, and compliance aspects in health-care edge environments presents an interesting research opportunity.
- Edge Computing for Smart Cities: Smart city applications heavily rely on IoT technologies. Exploring how edge computing can support and

enhance smart city services, such as intelligent transportation systems, environmental monitoring, energy management, and public safety, provides important research potential. Investigating data integration, analytics, and resource management in large-scale edge-enabled smart city deployments can contribute to more efficient and sustainable urban environments.

- Edge Robotics and Drones: Investigating the integration of edge computing with robotics and drones presents exciting research opportunities. Exploring how edge intelligence can enhance the autonomy, decision-making, and real-time capabilities of robots and drones can have an important impact on various fields, including logistics, surveillance, agriculture, and disaster response.
- Edge-to-Edge Collaboration and Federated Learning: Investigating collaborative approaches among edge devices to collectively process and analyze data presents interesting research avenues. Exploring distributed and federated learning techniques, where edge devices collaboratively train and improve machine learning models while preserving data privacy, can enable efficient and privacy-preserving data analytics in decentralized edge environments.
- Edge Security and Privacy: The ongoing research area of improving the security and privacy of edge environments opens avenues for future exploration. Subsequent research can concentrate on advancing intrusion detection and prevention mechanisms, devising secure data sharing and access control protocols, formulating privacy-preserving data analytics techniques, and incorporating blockchain-based solutions. These endeavors aim to fortify trust, security, and privacy within edge systems.
- Edge Computing for Edge AI Chipsets: Investigating hardware designs and architectures specifically tailored for edge AI applications is a research opportunity. Developing energy-efficient edge AI chipsets, specialized accelerators for edge analytics, and optimized hardware-software co-designs can improve the performance, energy efficiency, and scalability of edge computing platforms.
- Edge Computing in Industrial IoT: Edge computing holds essential huge potential for Industrial IoT (IIoT) applications, including smart factories and industrial automation. Research efforts can concentrate on creating edge-based solutions tailored for real-time data analytics, anomaly detection, predictive maintenance, and resource optimization in industrial

environments. Further exploration into integrating edge computing with Industrial Control Systems (ICS) and addressing the distinctive challenges related to IIoT security and reliability represents an important research direction.

- Edge Computing Economics and Business Models: Researching the economic aspects and business models of IoT-based edge environments is critical for successful adoption and sustainability. Exploring cost models, pricing strategies, revenue-sharing mechanisms, and value creation in edge ecosystems can provide information of the economic viability and commercialization of edge computing solutions.

In summary, these research opportunities provide exciting prospects for advancements in IoT-based edge environments and have the potential to shape this environment.

7.1 AI-IoT-Based Edge Environment

In today's era, integrating artificial intelligence (AI) with IoT-based edge environments offers better opportunities to augment the capabilities and intelligence of edge systems. Here are some key aspects and research directions related to AI-IoT-based edge environments:

- Edge-based AI Inference: Conducting AI inference at the edge has the potential to facilitate real-time decision-making and minimize latency by circumventing round-trips to the cloud. Research efforts can concentrate on formulating efficient AI inference algorithms and lightweight models suitable for deployment on resource-constrained edge devices. Techniques like model compression, quantization, and pruning can be explored to optimize both model size and computational requirements.
- Edge AI Model Training: Training AI models at the edge can use local data and reduce the need for data transmission to the cloud. Federated learning and edge-based distributed learning approaches can be explored to train models collaboratively across multiple edge devices while preserving data privacy. Research can investigate techniques for model synchronization, aggregation, and privacy-preserving learning in edge environments.
- Edge Intelligence and Adaptability: AI techniques have the potential to empower edge devices to learn and adapt to evolving environments and user preferences. Research can emphasize the development of edge intelligence algorithms capable of dynamically adjusting device behavior,

optimizing resource allocation, and personalizing user experiences. Techniques like reinforcement learning, transfer learning, and online learning can be explored to enable adaptive and autonomous edge devices.

- Edge Anomaly Detection and Predictive Maintenance: AI algorithms can be used to detect anomalies in data collected by IoT devices at the edge, enabling proactive maintenance and fault prediction. Research can investigate anomaly detection techniques, anomaly explanation models, and predictive maintenance strategies for edge environments. AI-based anomaly detection can help identify security breaches, equipment failures, and abnormal behavior in real-time.
- Edge-based Natural Language Processing (NLP): Enabling natural language understanding and processing at the edge can enhance user interactions with IoT devices. Research can focus on developing efficient NLP models that can run on edge devices to enable speech recognition, language translation, sentiment analysis, and voice-enabled control. Optimizing model size, energy consumption, and real-time processing capabilities are important issues.
- Edge-Based Computer Vision: AI-powered computer vision capabilities at the edge can enable applications such as object detection, tracking, and recognition. Research can focus on developing lightweight computer vision models and algorithms that can run efficiently on edge devices. Techniques like transfer learning, model compression, and hardware acceleration can be explored to enable real-time and low-power computer vision at the edge.
- Edge-Cloud Collaboration for AI: Investigating how AI models and computations can be dynamically offloaded and distributed between edge devices and cloud resources is an important research direction. Developing efficient edge-cloud collaboration frameworks, load balancing algorithms, and decision-making mechanisms to determine the optimal division of computation between the edge and the cloud can optimize resource utilization and enhance AI capabilities.
- Edge AI Security and Privacy: AI-IoT edge environments raise security and privacy issues. Research can focus on developing robust AI-based security mechanisms, such as anomaly detection for detecting malicious activities, adversarial attack detection, and secure AI model sharing. Addressing privacy challenges by exploring techniques such as

differential privacy and federated learning can also contribute to the secure and privacy-preserving operation of AI-IoT edge systems.
- Energy-Efficient AI at the Edge: Energy efficiency is critical for edge devices, especially those powered by batteries. Research can investigate energy-efficient AI algorithms, hardware architectures, and low-power computing techniques specifically tailored for edge devices. Optimizing AI model inference and training processes to reduce energy consumption and exploring energy harvesting techniques can contribute to sustainable and energy-efficient AI-IoT edge systems.
- Edge AI Governance and Ethical Issues: As AI becomes more pervasive in edge environments, research is needed to address governance, ethical issues, and accountability. This includes exploring transparency and explainability for providing efficient and scalable services to the end users.

7.2 Blockchain-IoT-Based Edge Environment

The integration of blockchain technology with IoT-based edge environments holds great potential for enhancing security, trust, and scalability. Here are some key aspects and research directions related to blockchain-IoT-based edge environments:

- Secure Device Identity and Authentication: Blockchain offers a decentralized and tamper-resistant system for overseeing device identities and ensuring secure authentication in IoT edge environments. Research can concentrate on crafting blockchain-based identity management solutions that facilitate seamless and trustworthy device onboarding, authentication, and access control at the edge.
- Data Integrity and Trust: Blockchain can ensure the integrity and immutability of IoT data in edge environments. Research can explore techniques for securely storing IoT data on the blockchain, verifying data integrity through distributed consensus mechanisms, and ensuring the authenticity and provenance of data collected from edge devices. Smart contracts can be used to enforce data integrity policies and automate data validation processes.
- Decentralized Data Sharing and Monetization: Blockchain has the potential to facilitate secure and decentralized data sharing among edge devices, opening avenues for data marketplaces and monetization opportunities. Research efforts can center around creating blockchain-

based protocols and incentive mechanisms that streamline peer-to-peer data sharing, enable data access control, and establish fair data monetization models in IoT-edge environments.

- Scalability and Performance: Blockchain scalability is a challenge in IoT-based edge environments due to the large number of devices and high data throughput requirements. Research can investigate techniques such as sharding, sidechains, and off-chain computations to improve the scalability and performance of blockchain in edge environments. Optimizing consensus algorithms and exploring lightweight blockchain implementations can also contribute to efficient and scalable edge blockchain systems.
- Privacy and Data Confidentiality: The inherent transparency of blockchain poses several challenges in maintaining privacy and data confidentiality within IoT-edge environments. Researcher can discuss about privacy-preserving techniques, including zero-knowledge proofs, secure multi-party computation, and encryption schemes, to safeguard sensitive IoT data while leveraging the advantages of blockchain. Hybrid approaches that integrate blockchain with other privacy-enhancing technologies can be explored to strike a balance between transparency and privacy.
- Trust and Reputation Management: Blockchain can provide a decentralized mechanism for establishing trust and managing reputations in IoT-based edge environments. Research can focus on developing reputation management systems, consensus-based trust models, and reputation scoring mechanisms to enable reliable interactions and collaborations among edge devices. Investigating mechanisms for detecting and mitigating Sybil attacks and collusion in edge blockchain systems is also important.
- Energy Efficiency and Resource Optimization: Energy efficiency is important in resource-constrained edge environments. Research can explore techniques to optimize the energy consumption of blockchain consensus algorithms, reduce storage and computational requirements, and use energy-efficient consensus protocols specifically tailored for edge blockchain systems. Exploring consensus mechanisms that strike a balance between security, performance, and energy efficiency is a research direction.

- Interoperability and Standardization: Achieving interoperability among various blockchain platforms and IoT devices poses a challenge in edge environments. Research efforts can concentrate on formulating interoperability frameworks, cross-chain communication protocols, and standardized data formats. These initiatives aim to facilitate smooth integration and communication across diverse edge devices and blockchain networks. Standardization endeavors can also tackle compatibility and data exchange challenges within heterogeneous edge blockchain ecosystems.
- Governance and Legal Issues: Blockchain-IoT edge environments raise governance and legal challenges. Research can investigate governance models, consensus mechanisms for decision-making, and legal frameworks for managing smart contracts, data ownership, and liability in decentralized edge systems. Exploring self-governance mechanisms and regulatory frameworks that align with the decentralized nature of blockchain can facilitate the adoption and operation of edge blockchain systems.

In last, in near future, research need to be focused on real-world deployments and use cases to validate the feasibility, performance, and practicality of blockchain-IoT edge solutions. We need to collaborate with industry partners, conducting field trials, and evaluating the economic viability and user acceptance of blockchain-enabled edge environments to provide efficient and accurate/scalable services to the end users/consumers in an IoT-based edge environment.

7.3 AI-blockchain-IoT-Based Edge Environment

The convergence of AI, blockchain, and IoT in edge environments provides exciting possibilities for creating intelligent, secure, and decentralized systems. Here are some research opportunities and key aspects to consider in the AI-blockchain-IoT-based edge environment:

- AI-Driven Smart Contracts: Integrating AI capabilities into smart contracts deployed on blockchain can enhance their functionality and adaptability. Research can explore techniques to enable AI-driven smart contracts that can autonomously execute and adjust contract terms based on real-time data collected from IoT devices at the edge. This can enable dynamic and intelligent decision-making within blockchain-based edge systems.

- Federated Learning with Blockchain: Federated learning, a collaborative approach to training AI models, can be combined with blockchain technology to address privacy issues in edge environments. Research can focus on developing federated learning frameworks that use blockchain for secure and decentralized model aggregation, participant reputation management, and incentive mechanisms for edge devices. This can enable privacy-preserving and collaborative learning in distributed edge environments.
- Edge Intelligence Marketplace: Blockchain can facilitate the creation of decentralized marketplaces for trading AI models, algorithms, and edge intelligence services. Research can explore the development of blockchain-based platforms where AI developers can securely share, sell, and deploy their AI models on edge devices. This can encourage innovation, collaboration, and monetization of AI capabilities in the edge environment.
- Secure and Trusted AI Data Marketplaces: Blockchain can provide a trust layer for AI data marketplaces in edge environments. Research can focus on developing blockchain-based mechanisms for ensuring data integrity, provenance, and fair exchange in AI data marketplaces. Smart contracts can be utilized to enforce data sharing agreements, protect data privacy, and enable transparent and auditable transactions between data providers and consumers.
- AI-Driven Anomaly Detection and Security: AI algorithms can be employed in conjunction with blockchain for enhanced anomaly detection and security in IoT edge environments. Research can explore the use of AI techniques, such as machine learning and deep learning, to detect and respond to anomalies, security threats, and cyber-attacks in real time. Blockchain can provide an immutable record of security-related events and enable secure and decentralized incident response mechanisms.
- AI-Enabled Blockchain Consensus: Blockchain consensus algorithms can benefit from AI techniques to improve efficiency, scalability, and energy consumption. Research can investigate the integration of AI algorithms, such as reinforcement learning, in consensus protocols to dynamically adapt to network conditions, optimize block creation and validation, and enhance the performance of blockchain networks in edge environments.

- Edge Device Reputation and Trust: Blockchain can serve as a decentralized mechanism for establishing and managing trust among edge devices in AI-based edge environments. Research can explore reputation systems based on blockchain to assess the reliability and trustworthiness of edge devices participating in AI computations and data sharing. Reputation-based trust models can enable secure and collaborative AI interactions in decentralized edge ecosystems.
- Privacy-Preserving AI on the Edge: AI models trained on sensitive data at the edge may raise privacy issues. Research can investigate techniques for privacy-preserving AI on the edge, such as secure multi-party computation, homomorphic encryption, and differential privacy. Combining these techniques with blockchain can provide a secure and auditable framework for privacy-preserving AI computations and data sharing in edge environments.
- Energy-Efficient AI-Blockchain-IoT Integration: Energy efficiency is so difficult to manage in edge environments. Research can focus on developing energy-efficient AI algorithms, lightweight blockchain protocols, and optimized integration frameworks for AI, blockchain, and IoT in edge systems. This can enable intelligent resource management, energy-aware computations, and efficient utilization of edge resources while maintaining desired levels of performance and reliability.
- Regulatory and Legal Issues: The integration of AI, blockchain, and IoT in edge environments raises regulatory and legal challenges. Research can explore legal frameworks, governance models, and compliance mechanisms

Hence, these are future research opportunities which we need to focus on, i.e., as our research work.

---

8 Research Statements for Security, Privacy, Trust, and Provenance Issues in IoT-Based Edge Environments

Here are some research statements related to security, privacy, trust, and provenance issues in IoT-based edge environments. We need to:

- Investigate innovative authentication and access control mechanisms to secure IoT-based edge environments, preventing unauthorized access to edge devices and data.

- Develop privacy-preserving techniques, such as differential privacy and secure multi-party computation, to protect sensitive data in IoT edge environments and mitigate privacy risks.
- Explore the application of blockchain technology to establish trust and accountability in IoT-based edge environments, ensuring the reliability and trustworthiness of edge devices and data transactions.
- Design lightweight and efficient encryption algorithms and protocols for secure data communication in resource-constrained IoT-edge devices, maintaining confidentiality and integrity.
- Investigate techniques for anomaly detection and intrusion prevention in IoT-edge environments to identify and respond to security threats in real time.
- Explore the integration of AI and machine-learning algorithms to enhance security and privacy in IoT-based edge environments, enabling proactive threat detection and mitigation.
- Develop decentralized and tamper-resistant data provenance mechanisms using blockchain technology to ensure data integrity, authenticity, and accountability in IoT-edge environments.
- Examine the impact of regulatory frameworks and compliance requirements on security, privacy, trust, and provenance in IoT-based edge environments and propose effective governance models.
- Investigate the role of edge intelligence and analytics in enhancing security, privacy, and trustworthiness of IoT-edge systems, enabling real-time threat detection and adaptive security measures.
- Explore the trade-off between security, privacy, and performance in IoT-based edge environments and developing optimization techniques to achieve a balance between these conflicting requirements.

Hence, these research statements provide a starting point for further investigation into the security, privacy, trust, and provenance challenges in IoT-based edge environments. Future researchers can focus more on specific aspects and propose innovative solutions to address these issues effectively.

---

9 Conclusion

This chapter discusses about several essential issues in IoT-based edge environments. The complexity and diversity of edge systems, combined with the substantial volume of sensitive data generated by IoT devices, necessitate robust measures to effectively address associated challenges. Upholding security is imperative, requiring the implementation of potent authentication mechanisms, secure communication protocols, and encryption techniques to safeguard data both in transit and at rest. Real-time identification and response to security threats can be facilitated through intrusion detection and prevention systems, alongside the deployment of anomaly detection algorithms. Additionally, access control mechanisms should be established to limit unauthorized access to edge devices and data.

Organizations and researchers must prioritize security, privacy, trust, and provenance issues throughout the design, development, and deployment of IoT-based edge systems. Maintaining a secure and privacy-preserving edge environment requires compliance with relevant regulations and standards, continuous monitoring, and proactive vulnerability management. Addressing these challenges necessitates interdisciplinary collaboration among researchers, industry experts, policymakers, and end-users. Establishing robust technical solutions, appropriate governance frameworks, and legal structures is essential to strike a balance between innovation and user rights, ensuring the creation of a trustworthy and secure IoT-based edge environment.

References

1. Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L.: Edge computing: vision and challenges. IEEE Internet of Things Journal. **3**(5), 637–646 (2016)
   [Crossref]
2. Satyanarayanan, M.: The emergence of edge computing. Computer. **50**(1), 30–39 (2017)
   [Crossref]
3. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of Things (IoT): a vision, architectural elements, and future directions. Future Generation Computer Systems. **29**(7), 1645–1660 (2013)
   [Crossref]
4. Roman, R., Zhou, J., Lopez, J.: On the features and challenges of security and privacy in distributed Internet of Things. Computer Networks. **57**(10), 2266–2279 (2013)
   [Crossref]

5. Kaur, H., Sood, S.K., Kaur, A.: Edge computing with IoT: frameworks, platforms, and applications. IEEE Access. **6**, 41047–41067 (2018)
6. Shi, W., Dustdar, S., Yao, W.: Mobile edge computing: vision and challenges. IEEE Internet Computing. **24**(5), 4–5 (2020)
7. Yi, S., Li, C., Li, Q., Wang, Q.: A survey of fog computing: concepts, applications and issues. Proceedings of the IEEE. **103**(1), 14–31 (2015)
8. Ning, H., Liu, K.: Future internet: the internet of things architecture, possible applications and key challenges. In: Internet of Things: Principles and Paradigms, pp. 123–163. Wiley (2018)
9. Mouradian, C., Naboulsi, D., Yangui, S., Glitho, R.H., Morrow, M.J.: A comprehensive survey on fog computing: state-of-the-art and research challenges. IEEE Communications Surveys & Tutorials. **20**(1), 416–464 (2018) [Crossref]
10. Bonomi, F., Milito, R., Zhu, J., Addepalli, S.: Fog computing and its role in the Internet of Things. In: Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, pp. 13–16 (2012) [Crossref]
11. George, T.T., Tyagi, A.K.: Reliable edge computing architectures for crowdsensing applications. In: International Conference on Computer Communication and Informatics (ICCCI), pp. 1–6 (2022). https://doi.org/10.1109/ICCCI54379.2022.9740791 [Crossref]
12. Sheth, H.S.K., Tyagi, A.K.: Mobile Cloud Computing: issues, applications and scope in COVID-19. In: Abraham, A., Gandhi, N., Hanne, T., Hong, T.P., Nogueira Rios, T., Ding, W. (eds.) Intelligent Systems Design and Applications. ISDA 2021 Lecture Notes in Networks and Systems, vol. 418. Springer, Cham (2022). https://doi.org/10.1007/978-3-030-96308-8_55 [Crossref]
13. Mishra, S., Tyagi, A.K.: The role of machine learning techniques in Internet of Things-based cloud applications. In: Pal, S., De, D., Buyya, R. (eds.) Artificial Intelligence-Based Internet of Things Systems. Internet of Things (Technology, Communications and Computing). Springer, Cham (2022). https://doi.org/10.1007/978-3-030-87059-1_4 [Crossref]
14. Nair, M.M., Tyagi, A.K.: Chapter 11: AI, IoT, blockchain, and cloud computing: the necessity of the future. In: Pandey, R., Goundar, S., Fatima, S. (eds.) Distributed Computing to Blockchain, pp. 189–206., ISBN:9780323961462. Academic. https://doi.org/10.1016/B978-0-323-96146-2.00001-2
15. Nair, M.M., Tyagi, A.K.: Privacy: history, statistics, policy, laws, preservation and threat analysis. Journal of Information Assurance & Security. **16**(1), 24–34 (2021) 11 p
16. Tyagi, A.K., Sreenath, N.: Fog and Edge Computing in navigation of intelligent transportation system. In: Intelligent Transportation Systems: Theory and Practice. Disruptive Technologies and Digital Transformations for Society 5.0. Springer, Singapore (2023). https://doi.org/10.1007/978-981-19-7622-3_7

[Crossref]

17. Aswathy, S.U., Tyagi, A.K., Kumari, S.: The future of Edge computing with Blockchain technology: possibility of threats, opportunities and challenges. In: Recent Trends in Blockchain for Information Systems Security and Privacy. CRC Press (2021)

OceanofPDF.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
S. Pal et al. (eds.), *IoT Edge Intelligence*, Internet of Things: Technology, Communications and Computing
https://doi.org/10.1007/978-3-031-58388-9_9

Secure Neural Network Inference for Edge Intelligence: Implications of Bandwidth and Energy Constraints

Jorit Prins[1] and Zoltán Ádám Mann[1]
(1) University of Amsterdam, Amsterdam, The Netherlands

Keywords Secure neural network inference – Privacy-preserving machine learning – Neural network – Edge intelligence – Energy consumption – Power consumption – Network bandwidth – Edge computing – Internet of Things

---

1 Introduction

With the rise in the availability of large amounts of data and computing resources, machine learning (ML) has gained huge importance. ML techniques like neural networks (NN) are promising ways to analyze data, make decisions, and predict future developments. ML has a wide variety of application domains such as water quality evaluation [32], mental health prediction [4], fake news detection [2], and more [5, 16]. The use of ML typically consists of two phases: training and inference. In the training phase, a NN is trained by feeding an extensive dataset to find the best parameter values for the NN. In the inference phase, the NN is applied to new inputs. The training phase is often a tedious and time-consuming process. Because not everyone has the time, resources, data, and know-how to train a NN, machine learning as a service (MLaaS) became popular [9]. In MLaaS, a company or other party offers a pretrained NN to its clients. This way, clients can benefit from inference with the NN without needing to worry about the training phase. This is an incredibly useful application in the Internet of Things (IoT) [29], for example, patient monitoring by medical sensors [28] or using IoT sensors to analyze real-time performance in automotive manufacturing [26].

A typical MLaaS situation (Fig. 1) consists of two parties: the client holding an input *x* and a service provider holding a pretrained neural network realizing a function *f*.

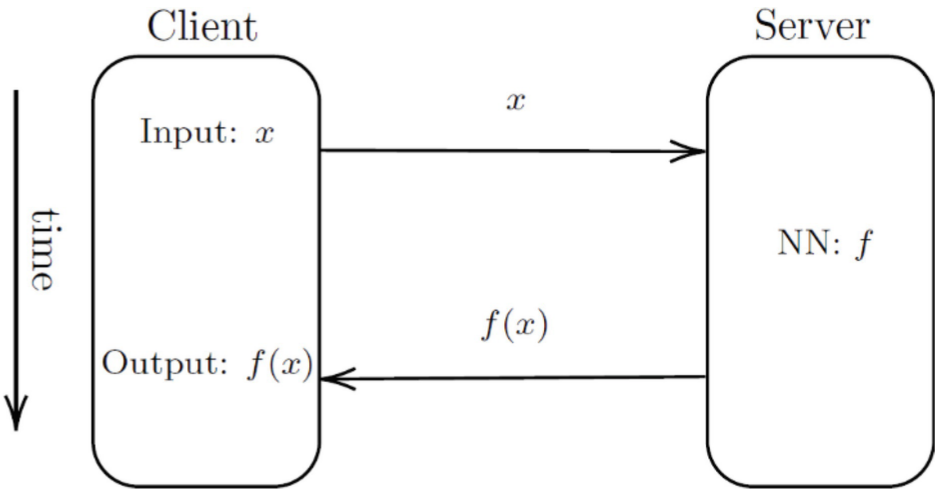


Fig. 1 Secure neural network inference (SNNI) process in a machine learning as a service (MLaaS) context. The neural network (NN) is already trained to realize a function *f*. The client provides an input *x* to the NN and wants to know the output *f*(*x*) of the NN for this input

This research focuses on the inference part: the client wants to know the output of the NN, available on a server owned by the service provider, applied to input *x* held by the client. This could be easily achieved: the client sends *x* to the server after which the server calculates *f*(*x*) and sends back the result to the client.

However, such a naïve implementation of MLaaS would lead to significant threats to security and privacy [30]. The client's input may be confidential, and the client may therefore be reluctant to send *x* to the server. Furthermore, the output *f*(*x*) of the NN on the given input could also be confidential, resulting in the need to retain this information from unauthorized parties. Besides, traditional security and privacy approaches may fail on IoT devices [27], for example, because these devices heavily rely on other nodes in the network [15].

An alternative implementation could consist of downloading the NN to the client and performing the inference there. However, this would also be problematic. For the service provider to train the model as accurately as possible, access to a large amount of precise data is needed, which may consist of sensitive information. This data, or properties of this data, could be stolen in the inference phase by the client. In addition, the service provider could be worried that the client or another adversary could steal the parameters of the NN, thus interfering with the business model of the service provider [22]. Lastly, this may not work for IoT devices since they are often power and resource constrained [15].

Therefore, the aim is that the client receives $f(x)$, without learning anything about the parameters of the NN (beyond what $f(x)$ might reveal about them), and the server does not learn anything about x or $f(x)$. This is the secure neural network inference (SNNI) problem. In recent years, many different approaches have been proposed to solve the SNNI problem [19].

The SNNI approaches proposed so far were mainly optimized for and evaluated in a setup where both the client and server computers are powerful machines, often in the cloud, connected by a network with high bandwidth. In addition, evaluation mostly focused on latency (i.e., the duration of the SNNI process).

However, there is an increasing tendency to employ ML inference in an edge computing setup and/or in connection with the Internet of Things (IoT), often involving sensitive data, like medical data [14, 28]. In such a setup, security and privacy requirements have to be considered in conjunction with requirements stemming from the constrained resources of the involved devices and of the network [17].

In edge computing and IoT settings, energy consumption also plays a crucially important role [1]. Client devices are often battery-powered, and their usefulness is heavily influenced by their battery life. Besides, client devices are often very constrained in terms of computing power. Companies, on the other hand, also want to keep energy consumption as low as possible because of energy costs and budget limitations and should therefore try to avoid big energy overhead. Moreover, a server is often connected to many clients in a deeply connected IoT network. Metrics like energy and power consumption are therefore also of importance and should be investigated. Another reason to limit energy consumption is the desire to reduce carbon emissions in the fight against climate change. For example, an estimation made in the SMARTer 2030 report is that all ICT systems worldwide will make up 2% of the global carbon emissions in 2030 [6]. The growing number of IoT devices also contributes to the increase in global power consumption. Researchers also state that ICT programmers will have the potential to avoid 20% of the global greenhouse gas emissions with smart programming. However, the energy consumption of SNNI approaches has not been investigated yet.

The aim of this chapter is to provide insight into the applicability of proposed SNNI approaches for edge intelligence in both edge computing and IoT setups. Our goal is to achieve a better understanding of the factors that could potentially limit the deployment of SNNI for edge intelligence in edge computing and IoT setups. We focus on two properties of such setups: (1) the importance of power and energy consumption, and (2) limited network bandwidth. We perform a set of experiments to investigate the power and energy consumption of two state-of-the-art SNNI approaches. In particular, we make the following contributions:

- We empirically compare the two SNNI approaches on two different NNs in terms of energy and power consumption on both the client and the server side.
- We investigate how power consumption changes over time during the SNNI process.
- We investigate the impact of the network bandwidth between client and server on SNNI latency and on power and energy consumption on both the client and the server sides.

The results that we report in this chapter offer many important insights into the applicability of existing SNNI approaches in edge or IoT setups, the factors for selecting one or the other SNNI approach, as well as pointers for future research, thus making progress toward secure edge intelligence.

The rest of the chapter is organized as follows. Section 2 introduces the necessary background knowledge. NNs, SNNI and the two approaches are discussed here. The logical design and technical setup, as well as the evaluation metrics of the experiments are discussed in Sect. 3. In Sect. 4, the initial measurements of both NNs are described and compared. In the second part, the relation between the bandwidth and the power consumption is explained. Section 5 presents the lessons learned, while Sect. 6 discusses related work, and Sect. 7 concludes the paper.

2 Preliminaries

This section summarizes the necessary background in neural networks, secure neural network inference, the CrypTFlow2 and Cheetah SNNI approaches, and the benchmark NNs that will be used later in this chapter.

2.1 Neural Networks

A feed-forward neural network (NN) computes a function, mapping an input vector to an output vector. The NN consists of a sequence of *layers*, numbered from 1 to L. The input of the first layer is the input of the NN, and the output of the last layer is the output of the NN. For $1 \le i \le L - 1$, the output of layer i serves as input of layer $i + 1$. Each layer computes a function, and the function computed by the NN is the composition of the functions computed by the individual layers.

Layers are often categorized as linear and nonlinear. Typical *linear layers* are:

- Fully connected (FC): computes the output vector by multiplying the input vector with a given weight matrix.
- Convolution (CONV): follows a more sophisticated definition, but can also be cast as multiplication of the input vector with a given matrix.
- Batch normalization (BN): scales each element of the input vector by multiplying it with a given number and adding another given number to the result.

Typical *nonlinear layers*:

- Activation functions: the same real function is applied to each element of the input vector. An often-used example is the ReLU function, which maps x to $\max(x, 0)$.
- Pooling functions: each element of the output vector is computed by applying a given function to a subset of the input numbers. An often-used example is max-pooling, in which the output number is the maximum of the considered input numbers.

A NN typically contains many parameters, such as the matrices in FC and CONV layers. The aim of the *training phase* is to find appropriate values for these parameters, so that the NN computes the desired function, or at least an approximation of it. For this purpose, the NN is evaluated on inputs with a known desired output. When comparing the actual output of the NN with the desired output, the parameters can be tuned such that the actual output of the NN becomes closer to the desired output.

During the *inference phase*, the NN is applied to new inputs for which no desired output is known.

In many cases, the NN is used for classification, i.e., to determine which of a given set of classes the input belongs to. In such cases, the quality of the trained NN can be quantified using its *accuracy*: the ratio of inputs in a validation dataset for which the NN outputs the correct class.

2.2 Secure Neural Network Inference

In machine learning as a service (MLaaS), a service provider has trained a NN and offers inference with this NN as a service. A client can use the service to obtain the output of the NN on a given input provided by the client.

Such a scenario may involve some secrecy constraints. The input and the output may constitute sensitive information that the client wants to keep secret. The parameters of the NN constitute the intellectual property of the service provider, which the service provider may not be willing to disclose. Thus, the secure neural network inference (SNNI) problem is to compute the output of the NN in such a way that the client only learns the output but nothing about the parameters of the NN (beyond what the output might reveal), while the service provider learns nothing about the input nor the output.

In recent years, several SNNI approaches have been proposed [19]. They use sophisticated cryptographic protocols to solve the SNNI problem.

In particular, SNNI can be cast as a secure 2-party computation (2PC) problem. 2PC means that two parties, typically denoted as Alice and Bob, compute $f(x, y)$, where f is a publicly known function, x is Alice's secret input, and y is Bob's secret input. A 2PC protocol guarantees that neither party learns anything about the other party's input beyond what the output reveals. In the case of SNNI, Alice is the client, Bob is the service provider, x is the input to the NN, y is the set of NN parameters (such as the elements of the weight matrices), and f is the evaluation of the NN with parameters y on input x.[1]

Several 2PC techniques exist that can be leveraged for SNNI. One such technique is *additive secret-sharing*. A number x is secretly shared between two parties by generating two numbers x_1 (given to one party) and x_2 (given to the other party) in such a way that $x_1 + x_2 = x$ but otherwise x_1 and x_2 are random. It is possible to perform simple operations, such as addition and multiplication, on secret-shared numbers using appropriate protocols. Such protocols can be composed to compute more complicated functions on secret-shared numbers.

Another useful 2PC primitive is *oblivious transfer* (OT). In OT, Alice has two messages m_0 and m_1, while Bob has a selection bit $b \in \{0, 1\}$. The aim is that Bob gets the message m_b, without learning anything about the other message, while Alice learns nothing. OT and its various extensions (e.g., for more than two messages) can be used as building blocks in many 2PC protocols. For example, the most efficient known protocols for computing some non-polynomial functions, such as ReLU, on secret-shared numbers, use OT.

Homomorphic encryption (HE) can also be used in the context of 2PC. An encryption scheme is called partially homomorphic if at least one operation on plaintexts (e.g., addition or multiplication) can be evaluated homomorphically, i.e., by appropriate manipulation of the corresponding ciphertexts. Fully homomorphic encryption (FHE) schemes support the homomorphic evaluation of both addition and multiplication. In a 2PC setting, FHE allows Alice to encrypt her input and send it to Bob, who can evaluate any polynomial function on Alice's encrypted input without learning Alice's secret input. After sending the encrypted result to Alice, she can decrypt it to obtain the result in plaintext.

Different 2PC protocols have different advantages and disadvantages. For example, homomorphic encryption is appropriate for evaluating linear layers of a NN, while for nonlinear layers, OT-based protocols are more appropriate [8]. Hence, many SNNI approaches combine different 2PC protocols on a per-layer basis. They use additive secret-sharing as an overarching scheme: as an invariant, the inputs to each layer are secret-shared between the parties, and the protocol for the layer yields its output again secret-shared between the parties. This way, different protocols can be composed, paving the way for a variety of SNNI approaches. Under appropriate conditions, it can be proven that such protocols satisfy the secrecy requirements [8].

2.3 CrypTFlow2/SCI$_{\text{HE}}$/SCI$_{\text{OT}}$

CrypTFlow2 [24] is a typical representative of SNNI approaches combining different types of protocols for different types of NN layers, using additive secret-sharing as the overarching method. The major contribution of CrypTFlow2 is a set of sophisticated and highly optimized protocols for nonlinear layers (ReLU, MaxPool, and ArgMax) and division, using OT. (Division is used to maintain a fixed bitlength in MeanPool layers.)

For linear layers, CrypTFlow2 implements two different protocols, one based on HE and another based on OT.

CrypTFlow2 ensures that its output is bitwise equal to the output of cleartext inference, i.e., the used security protocols do not distort the output in any way. For this reason, the SNNI approach of CrypTFlow2 is also called SCI, an abbreviation for Secure and Correct Inference. More precisely, CrypTFlow2 provides two SNNI approaches, denoted as SCI$_{\text{HE}}$ and SCI$_{\text{OT}}$. SCI$_{\text{HE}}$ and SCI$_{\text{OT}}$ only differ in whether they handle linear layers with HE or with OT.

In this work, we use SCI$_{\text{HE}}$ because it incurs less communication and is thus more efficient in the cases of limited bandwidth than SCI$_{\text{OT}}$ [24].

2.4 Cheetah

Cheetah [8] is one of the most recent SNNI approaches. It provides a highly optimized solution for SNNI. Cheetah was shown to be significantly faster than previous approaches, such as CrypTFlow2 [24], and able to perform secure inference with large neural networks such as ResNet50 in less than 2.5 min in WAN settings [8].

In line with most other recent SNNI approaches, Cheetah (1) uses additive secret-sharing as the overall 2PC scheme; (2) uses signed fixed-point arithmetic; (3) assumes that the NN architecture is known to both parties.

The main novelties of Cheetah are:

- For linear layers (fully connected, convolution, and batch normalization), Cheetah provides a new technique using homomorphic encryption. The used homomorphic encryption scheme is based on polynomials encoded as vectors. By arranging parameters carefully into the list of coefficients, homomorphic operations can be efficiently implemented as polynomial

multiplication. Additionally, in contrast to previous approaches, the used arithmetic is not in the ring $\mathbb{Z}_p$ (for a prime *p*), but in the ring $\mathbb{Z}_{2^k}$ (for a positive integer *k*), which makes implementation more efficient and also makes conversions between different rings unnecessary when switching between layers.
- For nonlinear layers, Cheetah uses improved versions of CrypTFlow2's protocols. For truncation (i.e., division by a power of 2), Cheetah allows a small error, enabling a significant speedup of the protocol. A new OT extension protocol (silent OT) is used in the protocols of both truncation and comparison, with the latter being the basis for multiple further protocols, such as for ReLU. The truncation protocol can be further accelerated if the most significant bit of the input is known, which is the case, for example, after a ReLU.
- Some further, smaller optimizations are introduced, for the special case of a convolution layer followed by a batch normalization layer, and for decreasing the amount of data transfer in the protocols based on homomorphic encryption.

Cheetah was experimentally evaluated using cloud servers with 2.70 GHz CPU and 16 GB RAM [8]. Two network setups were used in the evaluation: LAN with 384 MBps bandwidth and 0.3 ms latency, and WAN with 44 MBps bandwidth and 40 ms latency. Using multiple NNs (including SqueezeNet, ResNet32, ResNet50, and DenseNet121), Cheetah was compared to and found superior to CrypTFlow2. (For a fair comparison, the code in SCI_{HE} was modified to adopt the latest versions of the used libraries.) In addition, Cheetah was also compared to SecureQ8, a recent 3-party protocol. Cheetah proved faster than SecureQ8 in the WAN setting; however, in the LAN setting, SecureQ8 was faster than Cheetah.

2.5 Benchmark Neural Networks

In our experiments, we use two benchmark neural networks that were used in the past to evaluate CrypTFlow2 and Cheetah. Both of them are convolutional neural networks for image classification, but they significantly differ in terms of their size and their types of layers. The two networks are:

- SqueezeNet [11]: a network of moderate size that was specifically created to achieve relatively high accuracy with a strictly limited size (less than 500 thousand trainable parameters). It consists of:

  - 26 convolutional layers
  - 26 ReLU layers
  - 3 MaxPool layers
  - 1 AvgPool layer
- ResNet50 [10]: with over 23 million trainable parameters, ResNet50 is significantly larger than SqueezeNet. Moreover, ResNet50 features a richer set of layer types:

  - 53 convolutional layers
  - 49 batch normalization layers
  - 98 truncation layers
  - 49 ReLU layers
  - 1 fully-connected layer
  - 1 MaxPool layer
  - 1 AvgPool layer
  - 1 ArgMax layer

The protocols implementing secure inference with these two neural networks are readily available in both CrypTFlow2 and Cheetah.

---

3 Design of Experiments

In this section, we describe what we want to experimentally investigate and how we are going to do that. We first describe our experiments on the logical level and define the evaluation metrics, followed by the description of the technical setup used in the experiments.

3.1 Logical Design of the Experiments

Existing SNNI approaches like CrypTFlow2 and Cheetah were evaluated in a cloud environment, using powerful computers on both the server and client sides, and assuming a wired connection (either LAN or WAN) between the two computers. As evaluation metrics, accuracy, latency (i.e., the time needed to perform an inference), and the amount of transferred data were used.

In contrast, we are interested in the applicability of SNNI approaches in an edge computing or IoT environment. Applying SNNI in such an environment entails several challenges, of which we focus on two in this chapter:

- In edge computing, energy and power consumption are very important. Many edge devices are battery powered, which may limit both the momentarily power draw of the device and the available energy budget for the inference process.
- In edge computing, client and server machines may communicate over legacy wireless connections (e.g., 3G/4G), offering significantly lower bandwidth than what is available in typical wired LAN or WAN environments.

To investigate these aspects, we perform two sets of experiments. In the first set of experiments, we compare two SNNI approaches (CrypTFlow2 and Cheetah) using two NNs (SqueezeNet and ResNet50) in terms of energy consumption and power consumption. We measure energy consumption and power consumption separately for the client computer and the server computer, because in many edge computing scenarios, energy and power constraints are more stringent for the client than for the server. We also look at how power consumption changes over time during the inference process and how it varies between different executions of the inference process.

The second set of experiments focuses on the effect of the bandwidth available between client and server. For this purpose, we vary the available bandwidth and compare again the two SNNI approaches with the two NNs. We measure the duration of the inference process as well as total energy consumption and average power consumption on the client and on the server computer.

According to the experimental results of [8], Cheetah was clearly superior to CrypTFlow2 in every tested situation and according to all considered metrics. An interesting question is whether this holds true in the extended set of situations considered in our experiments and considering our extended set of metrics. The results might change the preference for choosing a specific SNNI approach in a given situation.

3.2 Evaluation Metrics

During our experiments, we collect the following metrics:

- Latency: the duration of the secure inference process for one input. This is measured as the difference in wall-clock time between the time when the client starts and when the client finishes its part in the inference process. Note that the server can start earlier than the client, but we disregard the time when the server is just waiting for the client since the actual protocol execution starts only when the client joins. At the end of the protocol server and client finish at about the same time. Unit: second (s).
- Average power consumption on the server side: the additional power consumption of the server computer caused by the SNNI program, averaged over the whole duration of the secure inference process. Note that only the duration of the protocol execution is taken into account, i.e., the setup time of the server before the client joins is disregarded. Unit: Watt (W).
- Average power consumption on the client side: analogously to the server-side power consumption, but measured on the client computer.
- Total energy consumption of the server side: the integral of the instantaneous power consumption over the whole duration of the secure inference process. In line with the above, only the duration of the protocol execution is taken into account, i.e., the setup time of the server before the client joins is disregarded. Unit: Joule (J).
- Total energy consumption on the client side: analogously to the server-side energy consumption, but measured on the client computer.

3.3 Technical Setup

For performing the experiments, we use two identical computers, one as server and one as client. Both computers have the following specification:

- CPU: Intel Xeon E-2378 @ 2.60 GHz, 8 cores
- RAM: 64 GB
- Network controller: Intel I350
- Operating system: Ubuntu 22.04.2 LTS (GNU/Linux 5.15.0-71-generic x86_64)

During the execution of our experiments, the two computers are used for no other purpose; thus, no other workload is running on them.

For measuring power and energy consumption, we use the Linux Hardware Monitoring (hwmon) interface.[2] Through this interface, we can obtain a new reading every second, giving the average power consumption of the system in the last second with a resolution of 1.0 W. By periodically measuring this value, we can calculate an approximation of the total energy consumption and the average power consumption of the system over a longer period of time.

The two computers are connected by a direct cable connection. For simulating different network bandwidths, we use the Linux tc (traffic control) utility to throttle the data rate of the client computer. For measuring bandwidth, we use the iperf tool (version 2.1.5).[3]

We use the latest version of Cheetah, which is commit 0b63d6f from March 2, 2023.[4] This codebase also includes a version of CrypTFlow2, which is also used in our experiments. We do not change any parameters of Cheetah or CrypTFlow2.

---

4 Experimental Results

In this section, we describe the results of our experiments. We start with some initial measurements to characterize the experimental environment in terms of bandwidth and idle power consumption. Then we conduct experiments on the power and energy consumption of Cheetah and CrypTFlow2 and report our findings in terms of both aggregated numbers (over the course of an inference) and the development of power consumption over time during the inference. Finally, we analyze the impact of throttling the network bandwidth on power and energy consumption.

4.1 Initial Measurements

We start our experiments by first measuring some basic properties of our experimental environment.

Without using any bandwidth throttling, the network connection between the client and server computers is characterized by the following parameters:

- Bandwidth between client and server: 941 Mbps
- Round-trip time between client and server: 0.6 ms

In addition, we measure the idle power consumption of both computers. For this purpose, we measure power consumption with a frequency of 1 s for a period of 10 min. From these 600 measurements, we compute the mean and standard deviation. The results are shown in Table 1.

Table 1 Idle power consumption of the test machines

	Mean	Standard deviation
Server	29.62 W	0.54 W
Client	29.73 W	0.68 W

From the low standard deviation, we conclude that the idle power consumption of both the server and client is fairly stable. Therefore, we can measure the power consumption of running a program by measuring the power consumption of the system while the program is running and subtracting from this value the mean idle power consumption of the system. We report this difference as power consumption, and also compute energy consumption on the basis of this difference in the experiments described next.

4.2 Power and Energy Comparison

In our first experiment, we use the experimental setup as described above, without any modification of the bandwidth. We measure the average power consumption and the total energy consumption of performing one secure inference with different SNNI solutions and different neural networks.

Table 2 presents an overview of the results. Each reported number is the result of averaging the metrics from 10 runs. The numbers under “average power” are the result of additionally averaging over the duration of the inference process.

Table 2 Comparison of performing one secure neural inference with Cheetah and $\mathrm{SCI}_{\mathrm{HE}}$ on two neural networks in terms of power and energy consumption. The shown numbers are the average of 10 measurements

Neural network	SNNI approach	Participant	Latency	Average power	Total energy
SqueezeNet	Cheetah	Client	25.5 s	29.7 W	772.1 J
		Server		27.91 W	725.7 J
	$\mathrm{SCI}_{\mathrm{HE}}$	Client	79.9 s	9.96 W	796.9 J
		Server		25.33 W	2026.57 J
ResNet50	Cheetah	Client	107.9 s	25.53 W	2757.61 J
		Server		42.31 W	4569.9 J

Neural network	SNNI approach	Participant	Latency	Average power	Total energy
	$\mathrm{SCI}_{\mathrm{HE}}$	Client	371 s	8.83 W	3276.33 J
		Server		45.12 W	16,741.35 J

In line with the results reported in [8], we can see that Cheetah is significantly faster than $\mathrm{SCI}_{\mathrm{HE}}$ on both neural networks. In light of this, it is also no surprise that the overall energy consumption (i.e., server and client together) of Cheetah is significantly less than that of $\mathrm{SCI}_{\mathrm{HE}}$.

However, investigating the server's and the client's energy consumption separately, we can see an interesting difference between Cheetah and $\mathrm{SCI}_{\mathrm{HE}}$. In the case of Cheetah, the distribution of energy consumption between client and server is balanced, whereas in $\mathrm{SCI}_{\mathrm{HE}}$, the energy consumption of the server is much larger than that of the client.

Investigating the average power consumption leads to even more interesting findings. In terms of server-side power consumption, Cheetah and $\mathrm{SCI}_{\mathrm{HE}}$ have roughly equal results. However, as a consequence of the large power imbalance of $\mathrm{SCI}_{\mathrm{HE}}$ between client and server, the client-side power consumption of $\mathrm{SCI}_{\mathrm{HE}}$ is significantly lower than that of Cheetah.

This leads to an interesting trade-off between client-side power consumption on the one hand and latency and overall energy consumption on the other hand. In a setup where client-side power consumption is not a major concern, Cheetah is more appropriate because of its lower latency and lower overall energy consumption. However, if client-side power consumption is a major limiting factor (especially for battery-powered client devices), then CrypTFlow2 is more appropriate. This is a new insight that was not visible in the experimental evaluation in existing work.

4.3 Timeseries Analysis

To obtain a more detailed understanding of the power consumption characteristics of SNNI, we now look at how power consumption evolves over time during the secure inference process.

The aim is to present the distribution of the power consumption at given points in time obtained from $k = 10$ measurements. A difficulty lies in the fact that the latency of the entire inference process is not constant across different runs. Therefore, we use the following methodology for determining the average power consumption at given points in time (see also Fig. 2):

1. We perform k runs, storing the power readings for every second in every run, leading to k time series. Let t_i denote the duration of the ith run.
2. We determine the average latency $t^* = \frac{1}{k}\sum_{i=1}^{k} t_i$.
3. We modify the time scale of each time series in such a way that their duration equals the average latency. That is, the time stamp of every power reading in a time series i is multiplied by t^*/t_i.
4. We split the interval $[0, \lceil t^* \rceil]$ into $\lceil t^* \rceil$ time slots with a size of 1 s.
5. For each of these time slots, we determine the average and the standard deviation of the measured power consumption values whose time stamp (after the modification of step 3) falls into the given time slot. This is the resulting time series that we visualize.

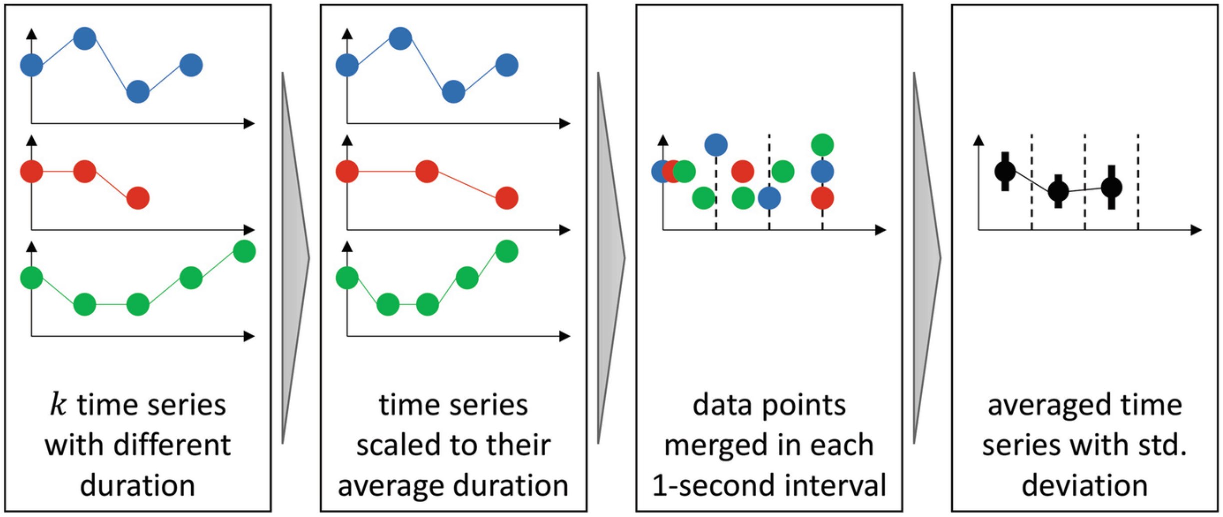

k time series with different duration
time series scaled to their average duration
data points merged in each 1-second interval
averaged time series with std. deviation

Fig. 2 Methodology for aggregating *k* time series with different duration

The results are shown in Fig. 3 for SqueezeNet and in Fig. 4 for ResNet50. A striking property of these plots is the large variance of the power consumption values over time. Short periods of low and high power consumption alternate quickly. In addition, when taking a closer look, it becomes apparent that in some periods the power consumption of the server and that of the client are closely correlated, whereas in other periods they are not.

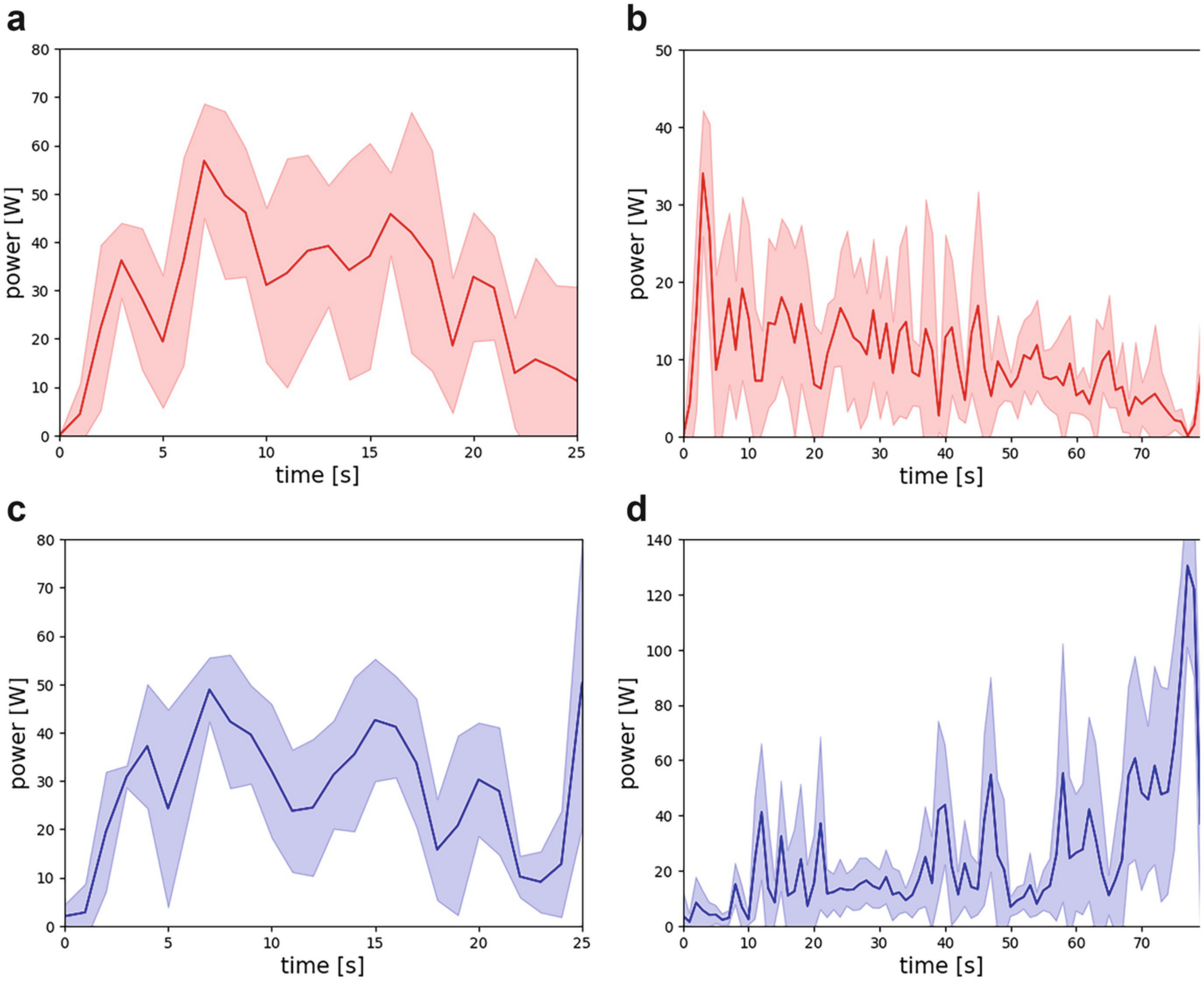


Fig. 3 Power consumption over time while performing secure inference with the SqueezeNet neural network. Each plot shows results distilled from 10 runs: the line shows the mean, while the shaded area shows the standard deviation around the mean. The plots on the left-hand side show results of Cheetah, while the plots on the right-hand side show results of $\mathrm{SCI}_{\mathrm{HE}}$, in both cases separately for the client (upper plot) and the server (lower plot). It should be noted that the scale of the axes is different in the different plots

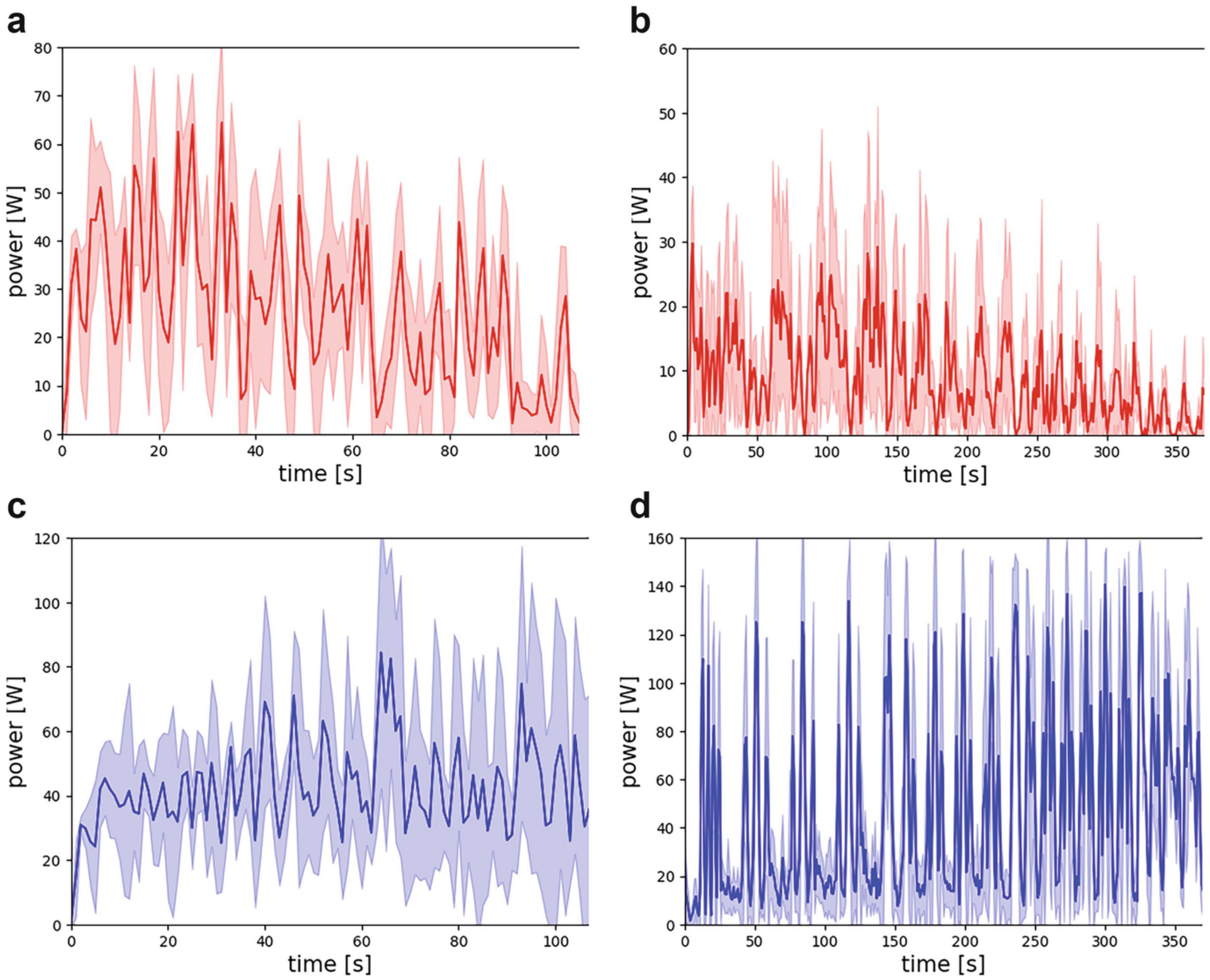


Fig. 4 Power consumption over time while performing secure inference with the ResNet50 neural network. The same remarks apply as in the case of Fig. 3

This seemingly strange phenomenon is actually not at all surprising. Remember that both CrypTFlow2 and Cheetah use different protocols for the different types of layers. The different protocols differ significantly in how much computation they require from the client and the server. Typically, linear and non-linear layers alternate, and the corresponding protocols are completely different in terms of being computation intensive or communication intensive. In some protocols, the load on server and client is similar, whereas in other protocols, the load distribution is strongly asymmetric. Also, layers of the same type can have different sizes, which can cause significant differences in power consumption.

4.4 Impact of the Bandwidth

By throttling the data rate of the client, we emulate computer networks with smaller and smaller bandwidths, and measure the impact on latency, average power consumption, and total energy consumption. Specifically, we perform measurements with the following bandwidth values:

- 941 Mbps (no throttling)
- 95.6 Mbps (result of specifying a throttling target of 100 Mbps)
- 19.2 Mbps (result of specifying a throttling target of 20 Mbps)
- 3.89 Mbps (result of specifying a throttling target of 4 Mbps).

We performed the experiments with both Cheetah and CrypTFlow2. However, based on the trend observed for the first three bandwidth values, we decided not to run CrypTFlow2 for the last (most constrained) bandwidth value. We had two reasons for this. First, the latency of CrypTFlow2 was increasing rapidly as the bandwidth was being reduced, which not only made our experiments last very long, but also made it clear that CrypTFlow2 was not practical in a system with this low bandwidth between client and server computer. Second, the average power consumption of CrypTFlow2 was decreasing quickly as the bandwidth was reduced, and it approached the range of measurement errors in our power readings (see the standard deviation in idle power measurement in Sect. 4.1), which made further power measurement unreliable.

The results are shown in Fig. 5. From subfigures (a) and (b), we can establish that with decreasing bandwidth between client and server, the latency of the secure inference process grows quickly for both SNNI solution approaches and for both NNs. This is not surprising, given the large amount of data transfer between client and server during the secure inference protocols. We can also observe that the latency difference between Cheetah and CrypTFlow2 (to Cheetah's advantage) grows with decreasing bandwidth. This is again no surprise since Cheetah causes less data transfer between client and server than CrypTFlow2, as shown in [8].

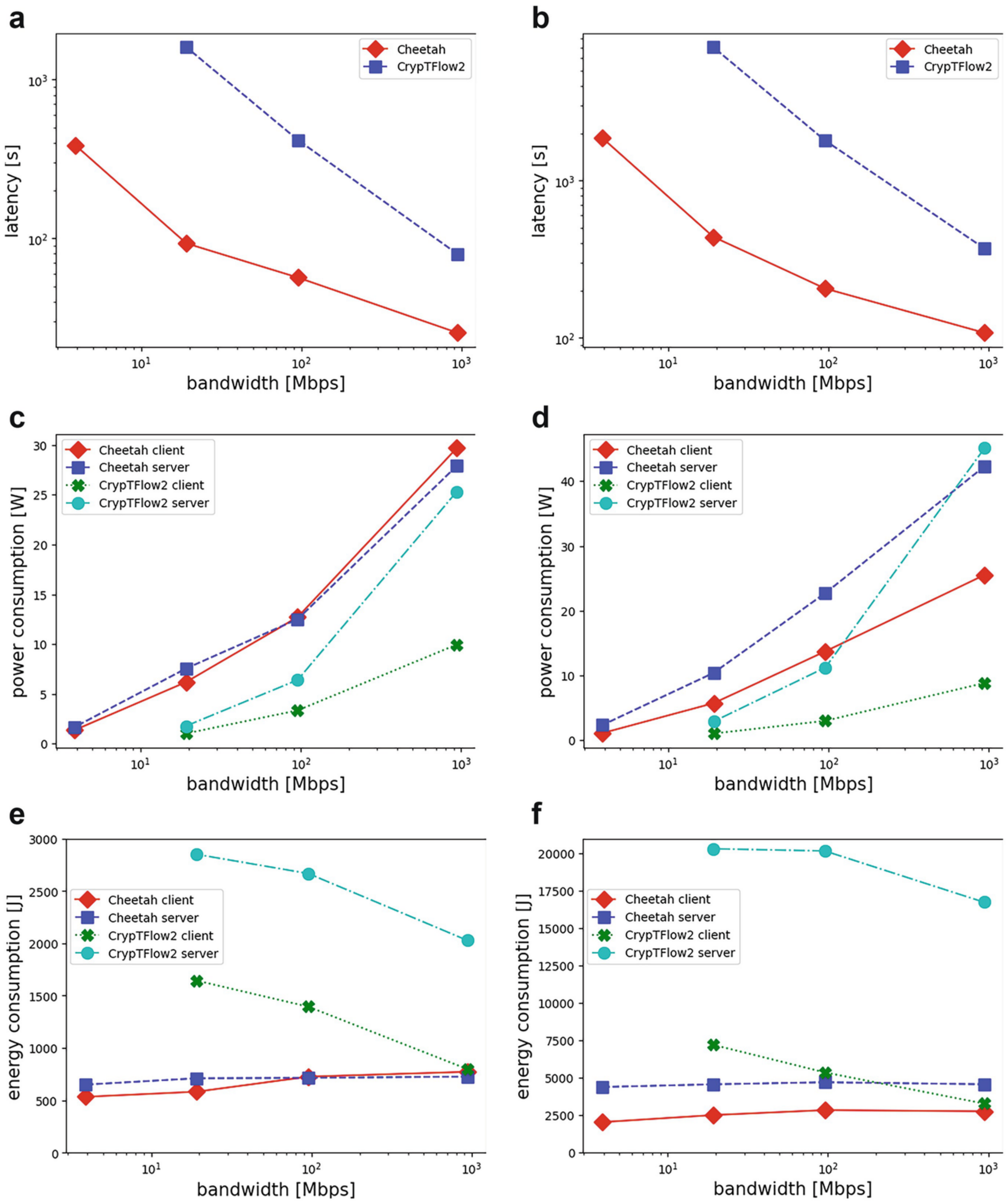


Fig. 5 Impact of the bandwidth between client and server on the latency, average power consumption, and total energy consumption of the secure inference process. Note the logarithmic scale of the horizontal axis in all plots; additionally, in the latency plots, also the vertical axes have a logarithmic scale. All reported numbers are the average from 10 runs. CrypTFlow2 was not run for the smallest considered bandwidth

The results on average power consumption (subfigures (c) and (d)) are more interesting. We can observe consistently across all measurements that reducing the bandwidth leads to a reduction in average power consumption. This is probably due to the decreased bandwidth slowing down the whole secure inference protocol: as the bandwidth decreases, both client and server spend more time waiting for I/O, effectively reducing the rate with which they can perform computation.

For similar reasons, since CrypTFlow2 has to transfer more data between client and server than Cheetah, it has a lower average power consumption than Cheetah. This is superposed by a higher asymmetry between client and server in terms of computation, and thus average power consumption, for CrypTFlow2 than for Cheetah. As a result, the advantage of CrypTFlow2 over Cheetah in terms of average power consumption is much more significant on the client side than on the server side.

The results for total energy consumption (subfigures (e) and (f)) are less conclusive. For CrypTFlow2, reducing the bandwidth between client and server seems to lead to higher total energy consumption, whereas for Cheetah, reducing the bandwidth seems to lead to no significant change or even to a slight reduction of total energy consumption. The reasons for this may require further, more in-depth analysis of the involved protocols. We can also observe that CrypTFlow2 is competitive with Cheetah in terms of client energy consumption, especially for high bandwidth values; however, in terms of server energy consumption, Cheetah has a significant advantage.

5 Discussion

In this section, we discuss the lessons learned from our experiments, the consequences for future research, and potential threats to the validity of our findings.

5.1 Lessons Learned

In the following, we distill our key findings from the experiments.

Energy and Power Consumption of SNNI Do Matter As we have seen, secure inference can lead to significant energy consumption and significant instantaneous power consumption. In an edge computing or IoT setup, this can be problematic. Different SNNI approaches have different characteristics in terms of total energy consumption and average power consumption. Thus, depending on the specific constraints of the edge / IoT environment, one or the other SNNI approach may be more appropriate. While research in the field of SNNI has so far focused mainly on reducing latency, constraints on power or energy consumption may be more stringent in some environments than constraints on latency.

Energy Consumption and Power Consumption Need to Be Considered Separately Although there is a clear connection between power consumption and energy consumption, low power consumption does not guarantee low energy consumption, and also low energy consumption does not guarantee low power consumption. For example, we have seen cases where CrypTFlow2 leads to lower average power consumption, but Cheetah leads to lower total energy consumption. Therefore, it is important to consider if, in a given target environment, average power consumption or total energy consumption is more important, and this may necessitate different design choices in the used SNNI protocols.

Client and Server Power Consumption Can Differ Significantly We have seen that, in terms of the balance between client-side and server-side power consumption, CrypTFlow2 is much more asymmetric than Cheetah, meaning that the CrypTFlow2 client consumes less power than the server, and also less than the Cheetah client. This can be an advantage for CrypTFlow2 in environments where the power draw on the client side is strictly constrained. This consideration yields a more nuanced picture about the advantages and disadvantages of different SNNI approaches, something that was not considered in previous evaluations such as in [8].

Power Consumption Varies Significantly Over Time The instantaneous power consumption fluctuates widely over time. This was consistently observed for both SNNI approaches, for both NNs, and for both client and server. The fluctuation is logical, given the varying types and sizes of layers that make up a NN, and the different protocols used for different layers. However, the strong fluctuations make it challenging to schedule such processes.

Network Bandwidth Has Major Impact We have found that the bandwidth between client and server significantly influences all considered metrics. Reducing the bandwidth leads to an increase in latency, a decrease in average power consumption, and can have different influences on total energy consumption. Bandwidth may also influence which SNNI approach performs best according to a given metric. In existing work, typically only a LAN and a WAN setup were tested, where even the WAN setup has relatively high bandwidth. Wireless networks used in many IoT and edge computing environments have lower bandwidth, which was typically not considered in the evaluation of existing works, thus potentially missing important insights into the suitability of different SNNI approaches for low-bandwidth environments.

5.2 Consequences for Future Research

Our findings have shown the importance of energy and power consumption in SNNI, as well as the large impact of the network bandwidth between client and server. These findings have important consequences for the design, implementation, and evaluation of SNNI approaches that should be taken into account in future work.

First of all, more research is needed to understand the exact *requirements* on energy and power consumption for secure inference in typical SNNI use cases. In particular, it is important to understand the relative importance of requirements concerning latency, power consumption on the client respectively the server side, and energy consumption on the client respectively the server side. Beside the relative importance of these metrics, it is also important to understand the acceptable ranges for these metrics, e.g., the maximum acceptable client-side power consumption in typical SNNI use cases in edge computing.

Second, more research is needed to *devise SNNI methods* optimized for low power consumption and/or low energy consumption and for limited bandwidth. This may include optimizing existing techniques for such environments, for example, by shifting some of the load from the client to the server to better support battery-powered clients. On the other hand, completely new methods may be needed to achieve improvements on all considered metrics at the same time [3].

Finally, we need *experimental evaluation and comparison* of proposed SNNI approaches in realistic settings—for example, with low bandwidth between client and server—paying attention to the metrics relevant in IoT or edge computing settings, like client-side energy consumption. That is, in contrast to the current practice of latency-focused evaluation in LAN and WAN environments, power consumption and energy consumption should also be considered as additional metrics, and bandwidth limitations of typical wireless networks should also be tested. A more realistic evaluation and comparison of SNNI approaches may lead to a preference for other methods than the one that performs best in terms of latency in a LAN or WAN environment.

In addition to these direct consequences of our findings, it is also worth investigating whether an *artificial cap on the bandwidth* between client and server can help reduce power consumption if needed. Our experimental results suggest that this is possible. Thus, if there is a strict limit on power consumption, artificially throttling the network bandwidth could help keep the power consumption low. This way, a trade-off between power consumption and latency could be controlled directly via the bandwidth.

5.3 Threats to Validity

We tried to perform our study carefully, but some threats to the validity of our study still remain. In the following, we review the most important threats to internal and external validity.

Internal Validity Our measurements of power consumption and energy consumption assumed that the used machines have a stable level of idle power consumption, and all additional power consumption can be attributed to the secure inference process. It is conceivable that some other programs may also create additional power consumption at some points in time, thus influencing the power consumption that we attribute to the secure inference process. To mitigate this threat, we observed the level of idle power consumption over a longer period of time, without experiencing significant deviations (see Sect. 4.1). In addition, all of the reported numbers are the average of 10 measurements, thus decreasing the implication of random effects.

Similarly, the way we measure latency also assumes that no other processes take a significant amount of time, compared to the latency of SNNI. Again, random effects, like the occasional activation of some system services, could impact the results. We mitigated this threat by averaging latency measurements over 10 runs. We did not observe major fluctuations in latency among the 10 runs in any of our experiments.

External Validity It is not clear to what extent our findings transfer to other setups. To improve generalization possibilities from our findings, we performed experiments with two SNNI approaches and with two NNs. However, it is possible that repeating our experiments with other SNNI approaches, other NNs, on other computers, and using other computer networks would yield significantly different results. Further experimental research could help establish an improved coverage of relevant setups. For example, it would be interesting to investigate the effects of a higher round-trip time between server and client, as well as scenarios in which multiple clients connect to the same server.

6 Related Work

In recent years, the SNNI problem has received considerable research attention, and several SNNI approaches have been proposed [19]. CryptoNets was probably the first approach to offer end-to-end protection in the NN inference process, although in a very limited setting and with huge overhead [5]. Subsequent work aimed mainly at reducing the overhead of SNNI. While the CryptoNets approach was based on homomorphic encryption, other work like DeepSecure used secure multi-party computation protocols, such as garbled circuits [23]. Gazelle combined homomorphic encryption and secure multi-party computation so that each layer of the NN can be evaluated with the most efficient protocol [12]. SecureML also used secure multi-party computation but proposed a different setup in which the actual protocol is carried out by two non-colluding servers, and the client only provides the input and collects the output [20]. Falcon used even three servers to further improve efficiency [31]. XONN used binary neural networks (i.e., NNs in which each weight, bias, and activation value is either 1 or -1) to enable more efficient secure multi-party protocols [25]. Delphi improved the protocols of Gazelle with several optimizations [18]. CrypTFlow created a framework for automatically turning TensorFlow code into a secure multi-party computation protocol implementation [13]. CrypTFlow2 extended the CrypTFlow framework with an efficient SNNI approach based on a combination of homomorphic encryption and different secure multi-party protocols [24]. Cheetah further optimized the protocols of CrypTFlow2 to yield one of the most efficient SNNI implementations to date [8].

All of the above works assumed a powerful client device with sufficient network bandwidth to the server. The proposed approaches were typically evaluated in a setting where both client and server were powerful cloud instances connected by a LAN or WAN. The constraints of typical edge computing or IoT setups were not considered. Also, power and energy consumption were not considered.

Some researchers investigated the SNNI problem in an edge computing, mobile computing, or IoT context. Instead of using compute-intensive cryptographic protocols, a possible approach that was suggested is to split the evaluation of the NN between client and server in such a way that the client evaluates the first couple of layers and sends the resulting features to the server, which then completes the inference [21]. However, this approach comes with no security guarantee, and the empirical experience showed that many layers have to be processed by the client in order to sufficiently constrain the leakage of information about the input to the server. Another proposed approach consisted of using secure multi-party computation using two edge servers [7]. In addition to assuming that the two edge servers do not collude, this approach also required a trusted third party to perform certain preprocessing tasks. A completely different approach is to perform the inference entirely on the client machine in a trusted execution environment [9]. However, this requires a relatively powerful client machine with support for the appropriate technology on the hardware level.

To summarize, the efforts culminating in CrypTFlow2 and Cheetah have created secure and efficient SNNI approaches, but failed to address the constraints of edge computing and IoT setups. On the other hand, existing work on SNNI for edge and IoT setups has serious limitations. Thus, there is a need to investigate paths for transferring Cheetah-type SNNI approaches to edge and IoT setups. Our work is a step in this direction. This work progresses further on the thesis of one of the authors [33].

7 Conclusion

In this chapter, we focused on some important aspects of edge intelligence by deploying SNNI in the context of edge computing or IoT applications: energy and power consumption, and the impact of the bandwidth between client and server. We used CrypTFlow2 and Cheetah, two state-of-the-art

SNNI approaches, and performed controlled experiments with them in a dedicated client-server environment.

Our experimental results revealed many interesting details. In particular:

- SNNI leads to significant energy consumption, with considerable differences depending on the used SNNI approach and NN.
- High energy consumption in SNNI does not necessarily mean high power consumption, and vice versa.
- Different SNNI approaches can differ significantly in terms of the balance between server-side power consumption and client-side power consumption.
- Power consumption of SNNI may vary considerably over time.
- Network bandwidth has a major impact on all considered metrics.

These insights can help choose the right SNNI approach in a given situation, considering the NN, power consumption constraints, network bandwidth, etc.

The gained insights also help inform future research into edge intelligence using SNNI in edge computing and IoT environments. In particular, we need more research to better understand the relevant requirements (in terms of power consumption, energy consumption, and bandwidth) in such environments, to devise SNNI methods specifically optimized for such environments, and to conduct more experimental evaluation and comparison of SNNI methods in such environments.

References

1. Ahvar, E., Ahvar, S., Mann, Z.Á., Crespi, N., Glitho, R., Garcia-Alfaro, J.: DECA: a dynamic energy cost and carbon emission-efficient application placement method for edge clouds. IEEE Access. **9**, 70192–70213 (2021) [Crossref]
2. Choudhury, D., Acharjee, T.: A novel approach to fake news detection in social networks using genetic algorithm applying machine learning classifiers. Multimed. Tools Appl. **82**(6), 9029–9045 (2023) [Crossref]
3. Chabal, D., Sapra, D., Mann, Z.Á.: On achieving privacy-preserving state-of-the-art edge intelligence. In: 4th AAAI Workshop on Privacy-Preserving Artificial Intelligence (PPAI-23) (2023)

4. Chung, J., Teo, J.: Mental health prediction using machine learning: taxonomy, applications, and challenges. Appl. Comput. Intell. Soft Comput. **2022**, Article ID 9970363 (2022)
5. Gilad-Bachrach, R., Dowlin, N., Laine, K., Lauter, K., Naehrig, M., Wernsing, J.: CryptoNets: applying neural networks to encrypted data with high throughput and accuracy. In: 33rd International Conference on Machine Learning, pp. 201–210. PMLR 48 (2016)
6. Global e-Sustainability Initiative (GeSI). #SMARTer2030—ICT solutions for 21st century challenges. Report. https://smarter2030.gesi.org/downloads/Full_report.pdf (2015)
7. Huang, K., Liu, X., Fu, S., Guo, D., Xu, M.: A lightweight privacy-preserving CNN feature extraction framework for mobile sensing. IEEE Trans. Dependable Secure Comput. **18**(3), 1441–1455 (2021)
8. Huang, Z., Lu, W., Hong, C., Ding, J.: Cheetah: lean and fast secure two-party deep neural network inference. In: 31st USENIX Security Symposium (USENIX Security 22), pp. 809–826 (2022)
9. Hou, J., Liu, H., Liu, Y., Wang, Y., Wan, P.J., Li, X.Y.: Model protection: real-time privacy-preserving inference service for model privacy at the edge. IEEE Trans. Dependable Secure Comput. **19**(6), 4270–4284 (2021)
[Crossref]
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
11. Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W.J., Keutzer, K.: SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size. arXiv preprint, arXiv:1602.07360 (2016)
12. Juvekar, C., Vaikuntanathan, V., Chandrakasan, A.: GAZELLE: a low latency framework for secure neural network inference. In: 27th USENIX Security Symposium (USENIX Security 18), pp. 1651–1669 (2018)
13. Kumar, N., Rathee, M., Chandran, N., Gupta, D., Rastogi, A., Sharma, R.: CrypTFlow: secure TensorFlow inference. In: IEEE Symposium on Security and Privacy (SP), pp. 336–353 (2020)
14. Lachner, C., Mann, Z.Á., Dustdar, S.: Towards understanding the adaptation space of AI-assisted data protection for video analytics at the edge. In: IEEE 41st International Conference on Distributed Computing Systems Workshops (ICDCSW), pp. 7–12 (2021)
15. Li, S., Xu, L.D., Zhao, S.: The internet of things: a survey. Inf. Syst. Front. **17**, 243–259 (2015)
[Crossref]
16. Li, Z., Yoon, J., Zhang, R., Rajabipour, F., Srubar III, W.V., Dabo, I., Radlińska, A.: Machine learning in concrete science: applications, challenges, and best practices. npj Comput. Mater. **8**, Article 127 (2022)
[Crossref]

17. Mann, Z.Á.: Security- and privacy-aware IoT application placement and user assignment. In: Computer Security—ESORICS 2021 International Workshops, pp. 296–316. Springer (2022) [Crossref]

18. Mishra, P., Lehmkuhl, R., Srinivasan, A., Zheng, W., Popa, R.A.: Delphi: a cryptographic inference service for neural networks. In: 29th USENIX Security Symposium, pp. 2505–2522 (2020)

19. Mann, Z.Á., Weinert, C., Chabal, D., Bos, J.W.: Towards practical secure neural network inference: the journey so far and the road ahead. ACM Comput. Surv. **56**(5), Article 117 https://dl.acm.org/doi/10.1145/3628446. (2023)

20. Mohassel, P., Zhang, Y.: SecureML: a system for scalable privacy-preserving machine learning. In: IEEE Symposium on Security and Privacy (SP), pp. 19–38 (2017)

21. Osia, S.A., Shamsabadi, A.S., Sajadmanesh, S., Taheri, A., Katevas, K., Rabiee, H.R., Lane, N.D., Haddadi, H.: A hybrid deep learning architecture for privacy-preserving mobile analytics. IEEE Internet Things J. **7**(5), 4505–4518 (2020) [Crossref]

22. Qayyum, A., Ijaz, A., Usama, M., Iqbal, W., Qadir, J., Elkhatib, Y., Al-Fuqaha, A.: Securing machine learning in the cloud: a systematic review of cloud machine learning security. Front. Big Data. **3**, 587139 (2020) [Crossref]

23. Rouhani, B.D., Riazi, M.S., Koushanfar, F.: DeepSecure: scalable provably-secure deep learning. In: Proceedings of the 55th Annual Design Automation Conference (DAC'18), p. art. 2 (2018)

24. Rathee, D., Rathee, M., Kumar, N., Chandran, N., Gupta, D., Rastogi, A., Sharma, R.: CrypTFlow2: practical 2-party secure inference. In: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS), pp. 325–342 (2020) [Crossref]

25. Riazi, M.S., Samragh, M., Chen, H., Laine, K., Lauter, K.E., Koushanfar, F.: XONN: XNOR-based oblivious deep neural network inference. In: USENIX Security Symposium, pp. 1501–1518 (2019)

26. Syafrudin, M., Alfian, G., Fitriyani, N.L., Rhee, J.: Performance analysis of IoT-based sensor, big data processing, and machine learning model for real-time monitoring system in automotive manufacturing. Sensors. **18**(9), 2946 (2018) [Crossref]

27. Sarker, I.H., Khan, A.I., Abushark, Y.B., Alsolami, F.: Internet of Things (IoT) security intelligence: a comprehensive overview, machine learning solutions and research directions. Mob. Netw. Appl. **28**(1), 296–312 (2023) [Crossref]

28. Somani, S., Russak, A.J., Richter, F., Zhao, S., Vaid, A., Chaudhry, F., De Freitas, J.K., Naik, N., Miotto, R., Nadkarni, G.N., Narula, J., Argulian, E., Glicksberg, B.S.: Deep learning and the electrocardiogram: review of the current state-of-the-art. EP Europace. **23**(8), 1179–1191 (2021) [Crossref]

29. Tran-Dang, H., Krommenacker, N., Charpentier, P., Kim, D.S.: Toward the Internet of Things for physical internet: perspectives and challenges. IEEE Internet Things J. **7**(6), 4711–4736 (2020)
[Crossref]
30. Timan, T., Mann, Z.: Data protection in the era of artificial intelligence: trends, existing solutions and recommendations for privacy-preserving technologies. In: The Elements of Big Data Value: Foundations of the Research and Innovation Ecosystem, pp. 153–175. Springer (2021)
[Crossref]
31. Wagh, S., Tople, S., Benhamouda, F., Kushilevitz, E., Mittal, P., Rabin, T.: Falcon: honest-majority maliciously secure framework for private deep learning. Proc. Priv. Enh. Technol. **2021**(1), 188–208 (2021)
32. Zhu, M., Wang, J., Yang, X., Zhang, Y., Zhang, L., Ren, H., Wu, B., Ye, L.: A review of the application of machine learning in water quality evaluation. Eco-Environ. Health. **1**(2), 107–116 (2022)
[Crossref]
33. Prins, J.: Energy Efficient Secure Neural Network Inference. BSc thesis, University of Amsterdam (2023)

Footnotes

[1] In this formulation of the SNNI problem, the structure of the NN is publicly known. A possible other formulation would entail that the structure of the NN is part of Bob’s secret input. However, we will stick to the formulation given above because it allows for more efficient implementations and is in line with the assumptions underlying most of the state-of-the-art SNNI approaches.

[2] https://www.kernel.org/doc/Documentation/hwmon/sysfs-interface

[3] https://sourceforge.net/projects/iperf2/

[4] https://github.com/Alibaba-Gemini-Lab/OpenCheetah/commit/- 0b63d6f2cfe979a446a7999ee78d705b6ef5ab81

OceanofPDF.com

Part III

Applications

OceanofPDF.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
S. Pal et al. (eds.), *IoT Edge Intelligence*, Internet of Things: Technology, Communications and Computing
https://doi.org/10.1007/978-3-031-58388-9_10

Internet of Things in Intelligent Transportation Systems

Ghalia Nassreddine[1] ✉, Amal El Arid[2] ✉ and Mohamad Nassereddine[3] ✉
(1) Department of Computer & Information Systems, Rafik Hariri University, Mechref, Lebanon
(2) Department of Electrical & Computer Engineering, Rafik Hariri University, Mechref, Lebanon
(3) Department of Electrical Engineering, University of Wollongong in Dubai, Dubai, UAE

✉ **Ghalia Nassreddine (Corresponding author)**
Email: nassreddinega@rhu.edu.lb

✉ **Amal El Arid**
Email: aridaa@rhu.edu.lb

✉ **Mohamad Nassereddine**
Email: mnassereddine@ieee.org

Keywords Intelligent transportation system – Internet of Things – smart roads – Smart streetlights – Smart parking

1 Introduction

The significant development of cities has recently become an important issue that involves governments, researchers, and industries. Indeed, during the information age, many people have left the village and come to live in cities that may offer a better life and more work opportunities. This urbanization creates new challenges for cities and governments such as huge power demand, pollution, huge mobility, inequality, and social exclusion [1]. These new challenges place pressure on urban infrastructures, particularly on transportation systems. Traditionally, cities have reacted to the change in mobility by extending the transportation supply by constructing new highways and roads. The huge traffic may lead to many problems like traffic jams, an increase in accident numbers, free parking, and many others. Recent technology may help governments in resolving such problems [2].

The huge development of information and communication technology (ICT) delivers new tools that may be used in resolving transportation problem. Recently, the integration of ICT with transportation systems leads to the creation of an intelligent transportation system (ITS). ITS is a diversity of contemporary information systems, networks, and electronic devices that are used and combined to manage and operate transportation and traffic. The goal of ITS is to enhance security, reduce traffic, and improve drivers' experiences [3].

Today, ITS applications are widely recognized and used in a variety of places. By giving information, such applications may improve road safety, reduce accidents, and control traffic congestion. ITS has evolved into a multidisciplinary, conjunctive field of study because to its boundless possibilities. As a result, various research institutions and enterprises around the world have developed ITS application solutions to suit the need. Furthermore, ITS can help reduce noise and air pollution, traffic, and energy consumption. The device monitors if there is enough space between vehicles to maintain a consistent speed, lowering fuel consumption and emissions. In addition, ITS gives drivers precise and timely information about traffic conditions and any potential road dangers, like roadblocks. ITS also allows the government to notify travelers of any changes or disruptions that may affect their journeys [3, 4]. Recently, new technologies such as artificial intelligence, machine learning, and the Internet of Things have been employed to improve the performance and efficacy of ITS [4, 5].

IoT is a recent transformation in the information technology sector. The name IoT or "Internet of Things" is composed of two main words [6]: Internet, which represents the global communication channel, and things, which represents any distinguishable object in the real world. Things are not only electronic devices. They can be personal, animal, furniture, or plants interconnected to this channel. Thus, things are real objects in this physical or material world. Therefore, IoT can be defined as a set of connected computing devices. Recently, many researchers defined the term IoT. However, the best definition is a wide and open network of intelligent objects that can auto-organize, exchange information and resources, and respond and act according to specific situations to make a change in the real environment [7, 8].

Nowadays, IoT is widely used in many sectors like medicine, power generation, renewable energy systems [9], home automation [10], smart cities [11], and intelligence transportation systems [12]. Intelligent transportation systems heavily depend on the IoT that assists in variety of tasks, such as improving data propagation, creating heterogeneous connectivity, and low-latency applications in high-capacity environments, cooperative transportation networks, and smart roadways [13].

The Internet of Things (IoT) technology can be used to solve the traffic problem in the most creative way. Vehicle-to-everything (V2X) is a network of connected devices made possible by the Internet of Things. This means that connected and autonomous vehicles using public roadways can exchange data concerning everything necessary to guarantee smooth traffic flow. The automobiles will communicate with other vehicles, signals, and traffic signs, thanks to sensors and other electronic devices placed in them. Based on this communication, the drivers can receive advice on how to adapt their speeds or lane choices to avoid entering the busy road. This transforms the stop-and-go traffic flow into an unbroken one, eliminating any possibility that a person may experience stress while driving [13].

This chapter describes the function of IoT in the intelligence. It is organized as follows: The usefulness of edge computing in ITS will be discussed in Sect. 2, the IoT in Sect. 3, the intelligent transportation systems in Sect. 4, the role of IoT in enhancing the performance of intelligence transportation systems in Sect. 5, and a conclusion about the importance of using IoT in the intelligence transportation system in Sect. 6.

2 Usefulness of Edge Computing in ITS

Edge computing (EC) is a distributed information technology structure where data is examined as near as conceivable to the network’s edge. Many scholars are currently focusing on ITS edge computing integration. Data-driven applications that benefit from the availability of big data to improve numerous aspects of our everyday lives, such as health, transportation, economics, and the environment, have been stimulated by the increased use of mobile devices and sensors. Recently, EC had the objective of addressing the restrictions of cloud computing, and ITS. It is a key component of contemporary cities. It is widely admitted that the incorporation of EC into ITS will significantly enhance the effectiveness, safety, and sustainability of transportation systems. In [76], the authors discussed the current developments in ITS sensing and EC applications. They focused on a critical aspect of ITS sensing. The main difficulties in ITS sensing and potential future approaches with edge computing integration were explored.

Autonomous vehicles, like self-driving automobiles, are made based on IoT devices and smart sensors that employ artificial intelligence (AI) approaches. Usually, these sensors are integrated with IoT devices to forward data in real-time mode. To achieve the necessary tasks based on sensor output data, IoT devices exchange available data with the processing unit. Using predefined instructions sent to the processor utilizing embedded and AI coding techniques, the processing unit executes the tasks. Fog computing, called also fogging technology, is a very helpful technique in lowering the amount of stream data burden on the cloud. It is a pre-processing stage of the data before deploying it into the cloud. In fog background, data optimization and analytical techniques are executed as part of data processing in a data hub on IoT devices or in a gateway [76].

To determinate the main track traffic circumstances, traffic management systems gather numerous video data and perform analyses of video processing. In [77], the authors proposed the use of edge computing to reduce bottlenecks that may appear due to the huge amount of collected videos. EC may deliver computing resources (like cloudlets) to edge nodes that are near to cameras. A cloudlet holds less computer power than TMC; therefore, it can only examine a small amount of video data.

In [78], the authors proposed a novel distributed multi-agent system-based edge-based AI-IoT integrated energy-efficient intelligent transport system for smart cities. Multiple regions and a finite number of zones were employed to present the division of an urban area. The right number of RSUs and edge computing hardware are set in a separate zone. Smart devices can gather a considerable amount of data from considerable infrastructures, devices, and sensors. The EC device can examine process, analyze, and predict using the raw data from the MAS that has been acquired. This predicted information can be helpful to freight vehicles to hold smooth and steady movement. Therefore, this situation may reduce GHG emissions and energy consumption.

---

3 Internet of Things

In 1999, Kevub Ashton used for the first time the term Internet of Things (IoT) to explain a system composed of many physical objects connected to the Internet via sensors [14]. Nowadays, IoT is widely used to represent techniques that employ internet connection to boost the computing ability of a set of things, machines, sensors, and everyday objects. The term "Internet of Things" is recently used. However, the idea of connecting computers and smart devices to observe and survey devices has been used for decades.

The Internet of Things (IoT) can be considered one of the most significant arising technologies. It may be categorized as preceding numerous other contemporary technologies, including artificial intelligence and machine learning [8]. Among all technologies that rise today, IoT will be essential to the future of many businesses [15, 16]. It is a network of connected things. Following that, the Internet will be made up of many different connected gadgets that will further expand the world's limits with physical items and virtual aspects. IoT will give the connected objects new powers [17].

In this section, the Internet of Things will be described. First, the different definitions are illustrated. After that, the general IoT architectures are presented. The main benefits of using IoT are illustrated later.

3.1 Definition

In the last decade, the IoT has gained considerable attention from many industries and researchers. Indeed, this attention comes from the capability of the IoT. It can provide a world in which small items can be connected and share information without human interaction [17]. The main objective of IoT technology is to improve human life. Certainly, through the utilization of IoT, individuals will find themselves encompassed by an array of intelligent devices capable of discerning human needs and preferences. These devices can work accordingly without direct orders [18]. There does not exist a unique definition for the term IoT. Table 1 summarizes the main existing definitions.

Table 1 Recent definitions of IoT in the literature

Reference paper	Font size and style
[19]	Dynamic and self-configuring global network architecture based on standards and interoperable communication protocols
[20]	An emerging paradigm that enhances human lives by enabling electrical devices and sensors to intercommunicate with each other via the internet. It employs the internet and smart systems and devices to deliver inventive solutions to businesses, governments, and public and private enterprises around the world
[21]	The set of interconnected hardware, software, network connectivity, and other electronic/computer components that offer networked devices/operations their responsiveness by allowing data collection and change
[22]	An emerging technology that allows the use of the internet to connect physical items or "things"
[23]	Related objects that have operating functions in what could be named the internet of the future
[24]	An open and complete network of intelligent objects that can auto-organize, exchange information and resources, react and act in the front of circumstances, and modifies in the environment
[25]	A network of a set of connected smart things that manage and access the data they collected or generated.

According to these definitions, the author defines IoT as a set of connected physical devices that communicate together to exchange data, manage data, make decisions, and act accordingly. The physical device can be any electronic device like a home device or industrial machine. By using suitable sensors and communication channels, these devices can deliver useful data and offer many services to people.

3.2 General Components of IoT

The main components of IoT are illustrated in Fig. 1 [26].

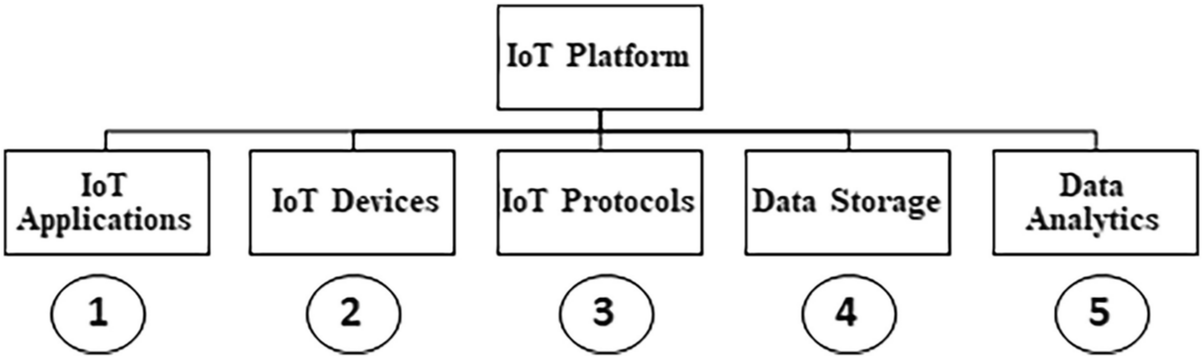


Fig. 1 The general components of the IoT platform

The first stage of creating an IoT application is to fix the application and choose the components like sensors, storage devices, and communication tools. Furthermore, the computation tools should be fixed to complete the main tasks of the IoT system. Suppose an IoT system for controlling cooling and heating system should be built. This system requires appropriate environmental sensors and communication channels between these sensors and the air conditioning devices [27]. The second component of the IoT platform is the IoT device. These devices could be data collection devices, processing devices, sensors, actuators, communication tools, or any other device that collects data. These devices can also function as IoT gateways, allowing data to be exchanged bidirectionally between devices and gateways. The third component of the IoT platform is the communication protocols. They can exchange information and data between the different components of the IoT platform and also with the controller for decision-making purposes.

IoT platforms deliver a wide variety of communication technologies and tools. Each of these technologies has specific features. The type of communication technology should be selected according to the IoT application. Cellular technology such as LTE-4G and 5G networks, Wi-Fi, and Bluetooth are examples of these technologies [28, 29].

The fourth component is the data storage of the IoT platform. It allows the gathering of data from IoT devices. Indeed, the amount of collected data is huge. Thus, there is a necessity to create efficient data storage like cloud servers. After storing the data, data analytics can be executed offline or in real time. It assists in decision-making and choosing the right action. It constitutes the fifth component of the IoT platforms.

3.3 General Architecture of IoT

IoT is a modern technology intended to facilitate communication between various components, such as sensors, through data storage utilizing the internet or other communication tools. IoT systems are created to facilitate daily human life. It is based on the use of smart devices, sensors, and communication channels to deliver innovative solutions to different problems and issues associated with diverse business, medical, and public/private industries across the world [20].

Growingly, IoT is becoming an essential element of human life that can be perceived everyplace around us. Indeed, IoT is an invention that links concurrently a vast variety of smart systems, platforms, intelligent devices, and sensors (see Fig.2).

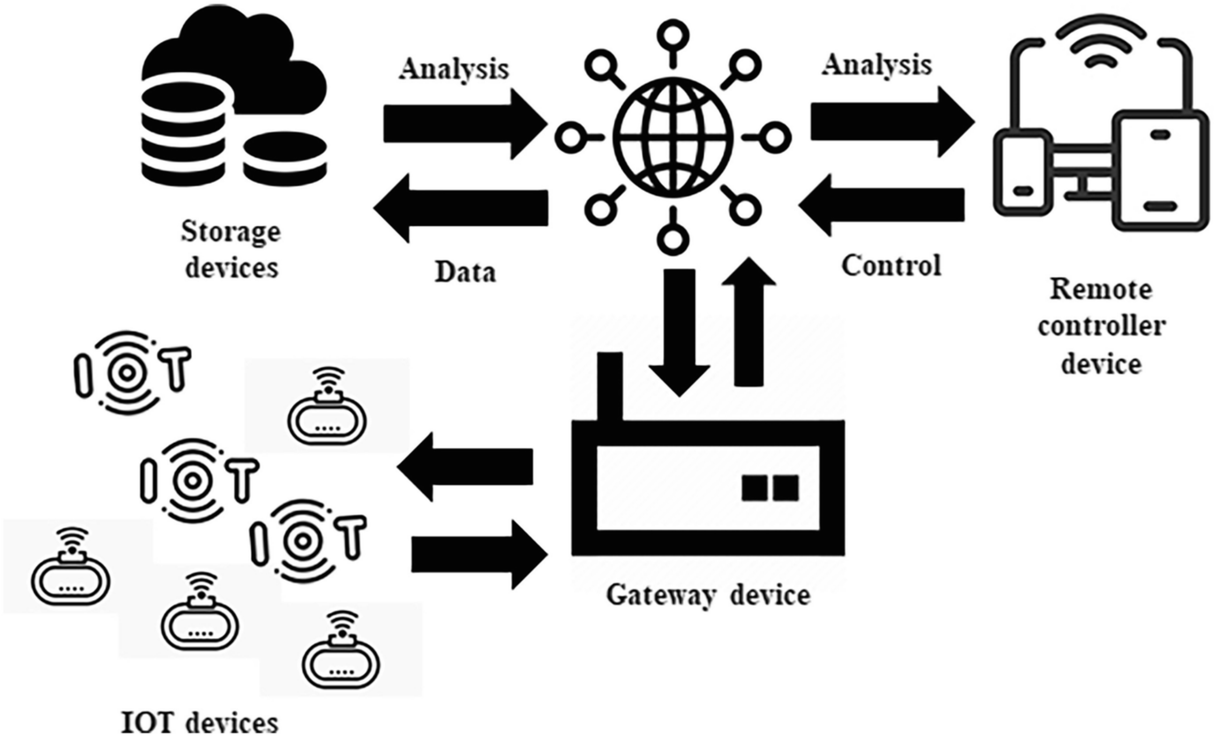


Fig. 2 General architecture of IoT systems

Additionally, it takes the benefit of quantum and nanotechnology for storage, sensing, and processing purposes [30]. Many kinds of research have been performed to demonstrate the probable significance and relevance of IoT transformations. It could be utilized as primary work before making novel innovative business plans while considering safety, warranty, and interoperability.

4 Intelligence Transportation Systems

In the last decade, road safety has become an essential sector of research for the automotive enterprise. It is one of the main ongoing interests of governments in the world. ITS is one of several possible solutions that can enhance road safety. A small group of transportation specialists developed the concept of ITS in the 1980s to identify the effect of computing and communications tools in the transportation area [31].

ITS can be defined as a set of smart systems that can perceive the environment, study the gathered data, and give the appropriate command to improve road safety. It contains varied information and communication tools that assist in exchanging, managing, and analyzing information to decrease traffic jams, enhance transportation control, minimize environmental damage, and improve the advantages of transportation to retail stakeholders and the people in general [32]. The solutions given by ITS are not limited to highway traffic, they also cover many other fields like rail systems and air-water transport systems.

This section is organized to describe the primary components of ITS, followed by illustrating the advantages of using ITS.

4.1 Components of ITS

Figure 3 shows the main components of ITS [33].

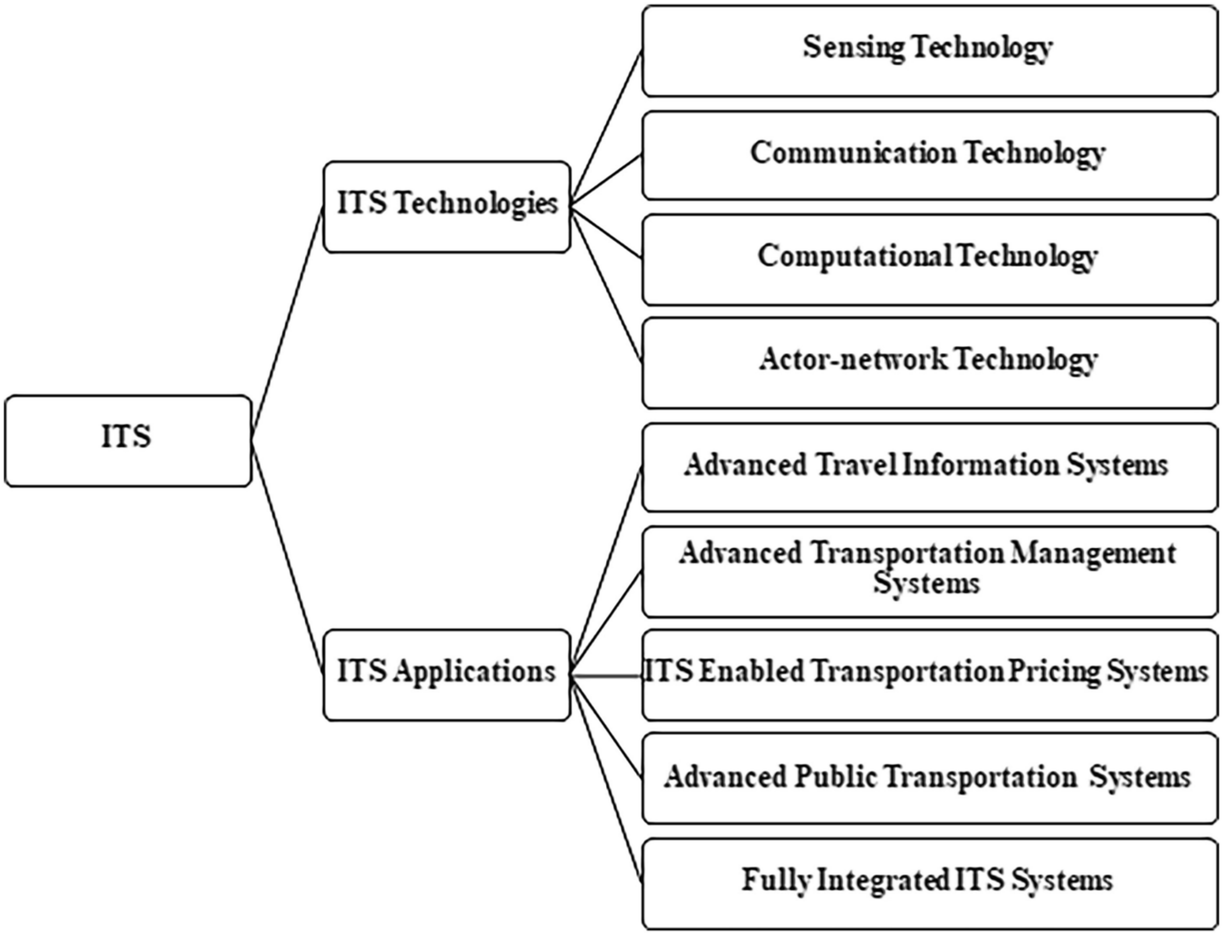

ITS
ITS Technologies
ITS Applications
Sensing Technology
Communication Technology
Computational Technology
Actor-network Technology
Advanced Travel Information Systems
Advanced Transportation Management Systems
ITS Enabled Transportation Pricing Systems
Advanced Public Transportation Systems
Fully Integrated ITS Systems

Fig. 3 The components of ITS technologies and applications

According to Fig. 3, the primary components of an ITS can be classified into two main sectors: ITS technologies and ITS applications. The components of technology can be grouped as follows:

(a) The sensors are used in the ITS to catch the adjacent vehicle traffic [34]. These sensors can be classified into:

- Active sensor: It is an electronic device that needs energy to work and is a radar instrument employed for calculating signals sent by the sensor that was reflected, twisted, or spread by the Earth's surface or its environment. Active sensors can collect data at any time and in any weather, as well as directly measure factors such as distance and velocity. Active sensors, on the other hand, have a low scanning speed and spatial resolution while being large and expensive.
- Passive sensor: It is a microwave device developed to accept and calculate raw emissions made by components of the Earth's surface

and its surroundings. The intensity calculated by passive sensors depends on the physical characteristics and temperature of the Earth. Passive sensors are sensitive to changing lighting and complicated settings.

(b) Communication technologies used on ITS are summarized as follows:

- Dedicated-Short Range Communications (DSRC) [35]: It is a technology that holds short-range to medium-range wireless communication. It works in the 5.8 or 5.9 GHz spectrum. It is particularly constructed for automotive services. Due to its soft latency, DSRC is appropriate for safety applications like post-crash reports.
- Wireless sensor network [36]: It is typically used for wireless Internet access. It permits quick vehicular communication. WiMAX is an example of this technology.
- Mobile telephony [37]: It is a famous technology, especially for metropolitan regions and main roads. However, it may be too slow; therefore, it is not appropriate for safety-critical ITS applications.
- Radio wave [38]: In Japan, real-time traffic information is transmitted through radio waves or infrared beacons along arterial roads, which are low-level service roads with a moderate capacity.
- Probe-vehicle [39]: In several nations, taxis and government-owned cars embedded with DSRC or other communication tools are frequently employed as probe cars to transfer required data like whereabouts and speeds. This data is received by a centrally controlled traffic management center (TMC) that performs an analysis step to create a picture of the area's traffic patterns and congested areas.

(c) Actor-network technology can perform an essential function in ITS to enhance vehicle protection. Indeed, the wireless system in a vehicle can be easily attacked if the vehicle is insecure or distributed. As a result, a VANET safety message must be protected to reduce false announcement from hostile vehicles and track the huge data produced by untrusted vehicles. A VANET (vehicular ad hoc network) is made up of groups of moving or stationary vehicles connected by a wireless network. A VANET credential or certificate should be offered to vehicles. However, these credentials should be revoked for hostile or untrusted vehicles. A hostile or untrusted vehicle is one that induces false or incorrect announcements. Three types of actors are explored for security applications that offer protection and privacy in VANETs, as will be indicated below [40]:

  - Vehicles or On-Board Units (OBUs): in this case, each vehicle can be considered an intelligent agent. It is provided with a tamperproof security device trustworthy for protecting sensitive and personal information [41].
  - Certification Authorities (CAs) define the trust structures. They deliver for each car a personal credential permitting it to confirm its identity during communication with other vehicles or transportation devices.
  - Road-Side Units (RSUs) that constitute a part of the VANET. Indeed, RSUs can hold a specific CA to perform specific security functionalities like creating public and private keys.

(d) Computational Intelligence (CI) represents the capability of a machine to learn from available data to perform specific tasks. Computational Intelligence has an important role in Intelligent Transport Systems. Floating Car Data (FCD) is one of the CI systems. It is a process to calculate the average traffic rate on roads according to speed, path, and time knowledge from conducting vehicles. This data constitutes an important source of traffic information for considerable intelligent transportation systems (ITS). Therefore, each equipped vehicle can be considered a moving sensor on the road network. Thus, the traffic flows can be determined and traffic announcements can be immediately developed [42]. Nowadays, two main types of FCD are on the road:

  - GPS-based FCD: In this type of car, a GPS receiver should be integrated within the car. The information gathered using car sensor is transmitted to the service provider via a typical onboard radio unit or via cellular network data. Accordingly, the exact location and direction of the vehicle can be determined [43].
  - Cellular Floating Car Data (CFCD) [44]: in this type, the data is emanated from cellular networks. Therefore, no specific device is

needed. The mobile phone’s location and movement are produced by a specific tools offered by the mobile network.

For the ITS applications, the following are identified:

(a) Traveler Information Systems: Traveler Information System is a crucial part of the transportation system. It helps travelers to recognize the advantages of transportation information systems. Nowadays, many travelers are continually using ITS for preparing their trips, selecting the best roads, scheduling their journeys, and choosing the transportation media. In-car navigation systems and telematics-based services that constitute a good tool for navigation, avoiding accidents, and guard services are part of this system [45].

(b) Transportation Management Systems (TMS): TMS’s primary goal is to provide control devices for traffic lights, ramp metering, and active message alerts on highways. These messages provide real-time information about traffic or highway status. Centralized traffic management centers, named Traffic Operations Centers (TOCs), perform these tasks. They are usually controlled by countries’ governments. This center merges several sensors and roadside devices, probe vehicles, alert signs, and other equipment to produce a full view of traffic flow and to detect vehicle wrecks or other unexpected events caused by severe weather or road dangers [46].

(c) Transportation pricing systems: In such systems, electronic toll collection (ETC) permits cars to automatically provide tolls based on a DSRC (Dedicated Short Range Communications). It allowed on-board equipment or ID to be displayed on the windshield. Similar technologies can also be used in High-Occupancy Toll (HOT) lanes that are reserved for buses and other large vehicles. In these lanes, single-occupant cars are only authorized after paying a fixed fee when there is a traffic delay. Transportation pricing systems may be able to control traffic flow in these dedicated lanes by applying variable pricing (through electronic toll collection) to maintain a smooth traffic flow even during rush hours [47].

(d) Public Transportation Systems (PTS) offers an AVL application that provides an automated location of the vehicle. This application is used for transit vehicles like trains to furnish statements about their current locations. Using this data, traffic operations managers can have real-time images of the status of transit vehicles. PTS improves the transit system by showing the arrival and departure times of buses and trains, providing clients with useful information such as the timing of the next train or next bus. PTS is also focused on electronic fare payment for public transit [48].

(e) Integrated ITS systems combine vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) to produce a compact platform. These systems can allow a mixture of driving services covering protection and non-protecting ITS applications. Dynamic lane, speed, and weather condition alerts are examples of such applications [49].

5 Internet of Things in Intelligent Transportation Systems

ITS systems include all systems that gather and interpret information to enhance the driving experience, minimize accidents, reduce traffic jams, and regulate traffic. Recently, using the Internet of Things (IoT), urban transport systems can become smarter, more secure, efficient, and use less energy. The essential result of the successful deployment of IoT is the optimization of the transportation system.

From a global perspective, ITS can be composed of four main parts: smart traffic management system, smart road, and smart parking. In this section, the role of IoT technology in these main parts will be illustrated.

5.1 IoT for Smart Monitor Traffic System

An intelligent Traffic Management system can be described as a smart system that can track vehicles and pedestrians to improve road safety. Figure 4 shows a general architecture of an intelligent traffic management system based on the Internet of Things (IoT).

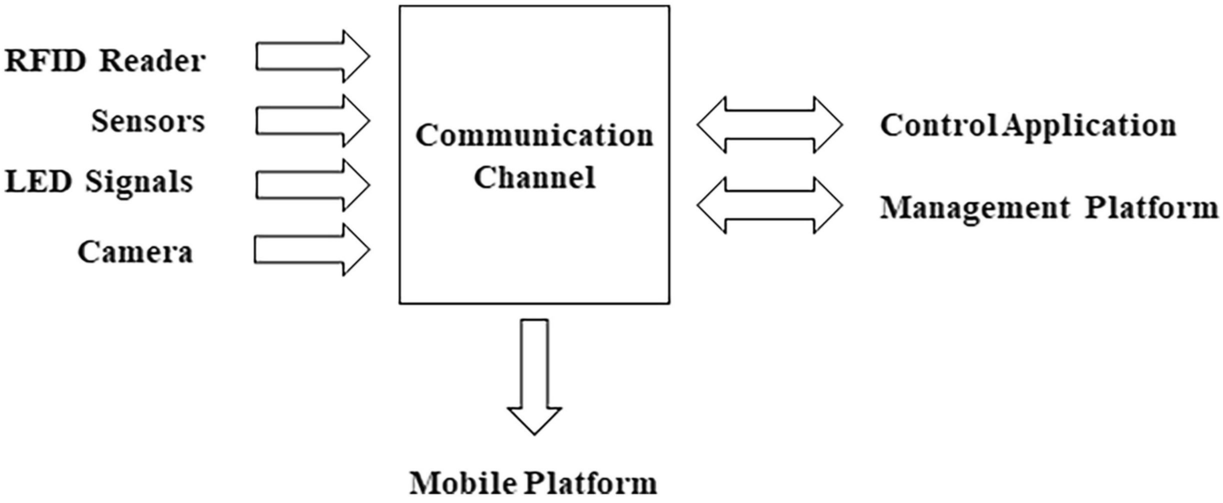


Fig. 4 IoT-based smart traffic management system

It is composed of four primary components [50]:

(a) Radiofrequency identification (RFID) is an electronic device used for sending the identity of an entity or an individual using wireless communication. It is one of the automatic identification technologies. RFID is appropriate for applications in various areas. Recently, it has been presented in many aspects of human lives. RFID is widely utilized in transportation to improve driving safety, prevent vehicle collisions, and even aid in the reduction of vehicle emissions [51].

(b) Sensors [52, 53] are categorized into:

- The motion sensor: surveys the speed of the vehicle to compute average vehicle speed and distance.
- The ultrasonic sensor: computes the distance to the destination by calculating the time between the emission and reception.
- PIR (Passive Infra-Red) sensor monitoring: witness's vehicles according to the emission or reflection of infrared radiation. They are employed to gather data from flow volume, vehicle presence, and occupancy.
- Speed sensor monitoring: used to estimate the time needed by the vehicle to reach the destination.

(c) LED lights are utilized as traffic lights. It is composed of a diode. When the p-n junction is forward-biased, it radiates light in the form of photons. We can get different colors; this color is determined by the gap in the energy band.

(d) A camera provides high-resolution pictures and video. These videos and images are transmitted to the smart devices to be managed and stored.

As illustrated in this figure, the components of IoT systems include a variety of sensors and other data-gathering tools. The collected data is managed and examined using a smart management platform. The managed information can be stored later on the server. This information is also used to monitor the transportation system. Useful information like road status can be shared with users via mobile or web platforms.

An Internet of Things and big data approach for the management of transportation systems was suggested by the authors in [54]. The network, data processing, data collection and acquisition, and application layers are further divided into four categories. Each layer is designed to handle data processing and management in a systematic way. Hadoop and Spark are utilized at the data processing layer to test the data. A wise community member can access the data by using the proposed event and decision mechanism based on named data networking. Several trustworthy sources' transportation datasets were used to analyze the proposed method. The results show how quickly data can be processed and distributed to citizens in real time. Findings from Spark in the Hadoop context are very accurate. The suggested architecture is significant in that it may be applied to a variety of automotive network scenarios.

To properly track weather-related information for traffic light regulation, heat- and moisture-sensitive sensors are also put in traffic lights. The traffic lights will get useful information from these sensors, and depending on the circumstances and the visibility of the road for moving vehicles, they will change the lights' brightness and intensity. To alleviate traffic congestion, real-time data feeds are employed in traffic lights. Sensors with Internet of Things capabilities put in strategic areas may collect data on high-traffic intersections and zones where cars are routed. Big data may be employed to further analyze this data, identify alternative routes, and enhance traffic signals to ease congestion. In addition, as the day turns into night, light dimming also happens when the weather gets worse. Roadside light sensors may be able to detect these signals and turn on and off the lights as needed [55]. The benefits of brightness control include energy savings and the prevention of accidents brought on by dazzling lights. IoT systems may also monitor and control traffic by automatically altering the lights at the right times using CCTV cameras with computer vision capabilities [56].

In Oslo, Norway, the first intelligent street lighting system was installed in 2006. Its goal was to regulate the streetlights' on/off cycles to conserve energy. Linked street lights, solar-powered street lights, motion-activated street lights, streetlights that serve as Wi-Fi access points, data-analytics-capable street lights, and ambient-controlled street lights are among the latest advancements in street lighting (see Fig. 5).

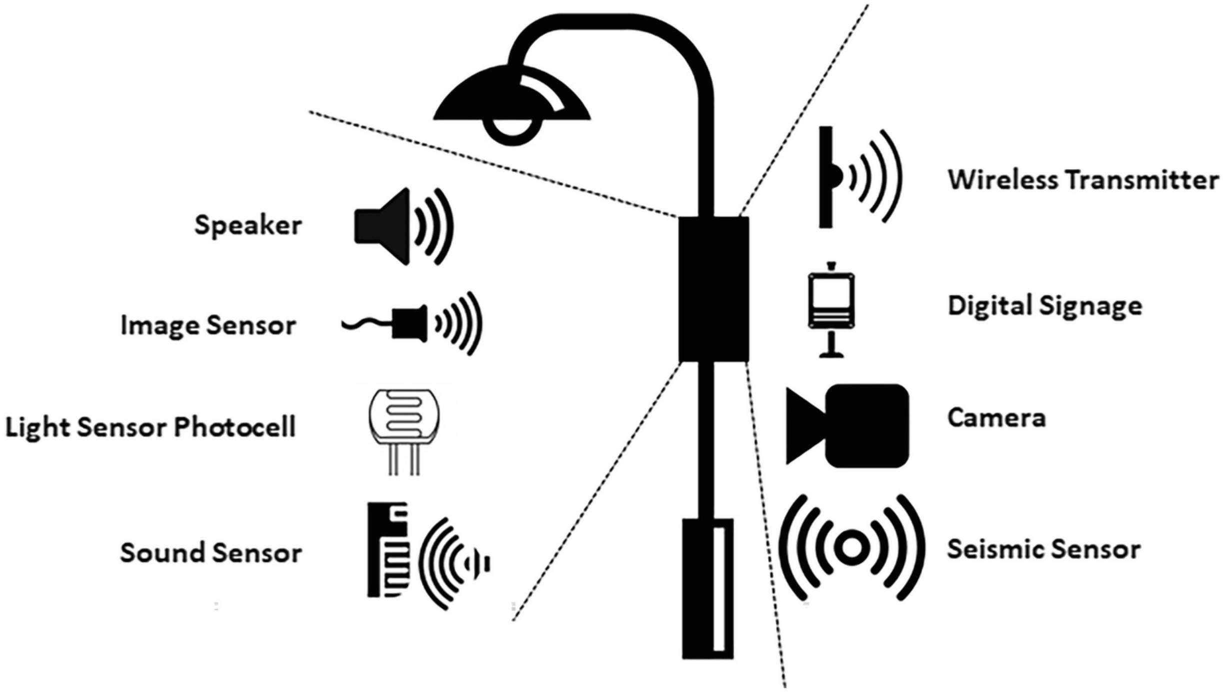


Fig. 5 Smart Street light system

Many monitoring applications can be carried out and integrated into smart city IoT networks by adding various sensors to streetlights, including detecting terrorists attack, monitoring air quality, and creating EV charging points for roadside parking monitoring.

In [57], the authors proposed an IoT-based smart and adaptive lighting in street lights. The proposed system can be utilized for intelligent, weather-adaptive street lighting. It is based on an intelligent embedded system that enables street light control based on sunlight detection. The street light automatically turns off at night. The ON/OFF can be accessed online at any time and from any location. Monitoring the activities on the road requires a camera that is mounted on top of the street light. The video footage is then recorded on a server. Additionally, a panic button is mounted on the pole. When pressed in an emergency or when someone is in danger, the panic button alerts the local police station.

Some researchers proposed an IoT/IoE method to improving transportation and mobility in smart cities [58]. Sii-Mobility, a nationwide smart city industry centered on mobility and transportation, was demonstrated to be an adjustable platform. Complex, heterogeneous IoT and IoE scenarios can be created by combining sensors, actuators, and other IoE components into a larger big data, machine learning, and data analytics scenario. A complete and complex case study has been delivered to validate the process in the context of a system that dynamically reverses the direction of travel of a road segment with all safety precautions in place. To show the viability of a flexible configuration that offers security, safety, local, cloud, and other hybrid solutions, this case study integrates several IoT platform-building components.

Every year, millions of individuals lose their lives in road accidents. The majority of these incidents are minor ones, yet the delay in providing emergency services still results in a significant loss of life. In order to provide prompt aid, a system that can analyze a situation and identify it as an accident would be quite beneficial. In [59], the authors reviewed different approaches employed to detect accidents, such as conventional accident detection techniques. Every moving object is regarded as a node in such approaches. The alert messages are transmitted via an RF module, and moving cars that are within range of the module can receive the alarm signals. The message is finally received by the vehicle in the network area, after which it is transmitted to the base station. There are four different message types in the alert message. Piezoelectric sensors, MEMS sensors (micro-electro-mechanical systems), flame sensors, and temperature sensors are used to detect them. The system quickly locates vehicle accidents and notifies the rescue crews of the incident’s location. Additionally, a switch has been added to the system so that the driver can instruct it to stop delivering alert messages if there are only minor injuries or losses. The controller receives data from the sensors, notifies the roadside unit about the accident, and then sends a message to the rescue crew. GPS and Wi-Fi are utilized to locate the vehicle. GSM modules in earlier systems were unable to interact in non-network areas, but they now offer coverage in both networks and non-network areas. In the event of minor damage, a switch is also accessible to cease message transmission.

For VANETs, the authors in [60] performed an overview of various congestion control methods. Three basic approaches—proactive, reactive, and hybrid congestion control strategies—are explored for dealing with the network congestion problem in VANETs. Latency and throughput are two crucial factors in addressing network congestion. The six main types of approaches are rate-based, power-based, hybrid, CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance), prioritizing, and clustering-based. The talks demonstrate that proactive techniques are best for managing network congestion in VANETs because they take proactive action. Priority-based approaches also exhibit superior performance when used in conjunction with hybrid congestion management techniques.

In the present day, motorcycles, commonly referred to as two-wheelers or motorbikes, are frequently utilized due to their affordability compared to other vehicle types. However, they are also considered more hazardous than traditional vehicles. Accidents can occur as a result of speeding or intoxicated driving. Everyone’s top priority when traveling by car is their safety and security. With the spectacular growth of transportation networks like two-wheeled vehicles and the fast urbanization, safety on the roads and security on bikes have become an unavoidable priority for us. Accidents now happen more frequently, causing more harm and more fatalities. In many cases, we are unable to determine where the accident occurred. An item of protective clothing worn to shield the head from harm is a helmet. The helmet primarily helps the skull protect the brain. A smart helmet can locate where an accident occurred, saving lives and making two-wheeler driving safer than before. In order to prevent the accident, the authors in [61] suggested a smart helmet system. The system is divided into three sections: the mobile application, the helmet circuit, and the automotive circuit. The helmet circuit initially includes an IR and alcohol detection sensor. A three-axis accelerometer, Bluetooth module, relay, and load sensor are included in the car’s circuit. If the helmet is worn out and no alcohol is detected, the helmet circuit signals the car circuit to commence. The auto circuit then verifies the load’s state before starting. A three-axis accelerometer detects collision or impact. When a mobile application detects an accident, it automatically notifies the police and emergency contacts of the accident's location via the database.

5.2 IoT in Smart Roads

Smart roads allow the detection of vehicles and the fast management of traffic signals. Smart roads can be built using IoT and a set of specific sensors. They produce a secure and soft driving experience by delivering real-time information to drivers, such as traffic, parking, road status, and the shortest path to the destination [62].

Smart roads are implemented based on new technology, such as IoT (see Fig. 6). IoT has the greatest potential for road infrastructures, and that could result in the most useful applications.

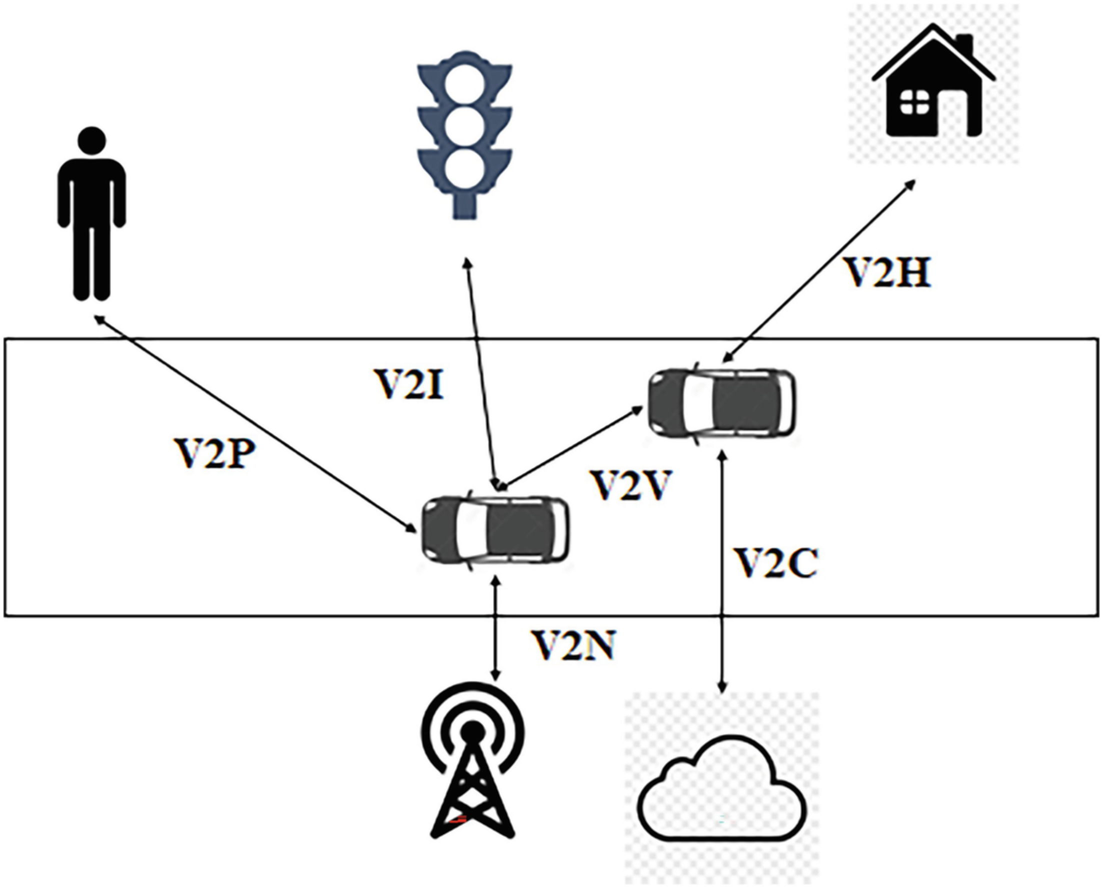


Fig. 6 Types of vehicle communication using IoT

Car-to-car communications and vehicle ad hoc networks are important for smart roads. V2N communications encompass alert messages, short notifications, hazard warnings dissemination, and collision avoidance transmissions from a vehicle to any roadside infrastructure. V2X communications that exist between a car and any other object (referred to as X) can integrate V2V, V2I, V2E (environment), V2P (pedestrian), V2C (cloud), and V2H (home) communications. There is no need for a cellular connection with V2X.

V2X is significant because it enables cars and objects to interact and exchange critical information, such as position, identity, physical presence, and speed. Cars can be notified about potential forthcoming traffic risks using such data, averting accidents and improving road user safety [63]. Table 2 summarizes some existing technologies that provide V2X communication [64–67].

Table 2 Existing V2X devices

Commercial devices	Cohda Wireless MK5 OBU	LocoMate OBU	LocoMate RSU	LocoMate ME
RSU/OBU/Mobile	OBU	OBU	RSU	Mobile
Supported V2X technologies	DSRC	DSRC	DSRC	DSRC
GNSS	Yes	Yes	Yes	Yes
Baseband processor chipset	u-blox THEO-P	Atheros AR5414	Atheros AR5414	–
Operating system	Linux	Linux, SDK with C libraries	Linux, SDK with C libraries	Linux, SDK with C libraries, android
External connectors	Ethernet, USB, CAN, audio, microSD	Ethernet, USB	Ethernet	–

Nowadays, many researchers and industries work on projects that may enhance road safety, for instance, in [68], a P-VERSE strategy to prioritize the emergency vehicles at a multiway intersection when there is no traffic signal was discussed. Emergency vehicles, such as ambulances, fire trucks, and police cars, were prioritized, and a clear path to travel was given. The traffic signal is a tried-and-true method for reducing accidents, avoiding vehicle collisions, and minimizing traffic jams. However, this causes vehicles to be stopped repeatedly, lengthening the time they must wait. A two-stage plan was created to avoid this problem:

(a) Markov chain to forecast each vehicle node's future course in the first phase

(b) A cooperative coalition-based game-theoretic technique for generating tactical interactions between vehicle nodes

These moving nodes operate as agents in the creation of coalitions. Depending on how useful the alliance is, the vehicular nodes either join or quit it. To protect the safety of the rescue vehicles, the nodes create a coalition.

The authors of [69] devised a novel solution to the traffic congestion problem based on IoT and data analytics. The proposed framework is made up of four layers. The first layer, or sensing layer, makes use of many data sources to ensure a reliable and precise measurement of traffic conditions on the streets. This data is then transmitted to the second layer. These data are used to make effective decisions by the second layer, or fog layer, and are also delivered to the third layer. At the third tier, the cloud layer, these data are permanently kept for analytics and knowledge discovery. The fourth tier, the services layer, provides traffic control support services. The scientists also presented a level 1 smart traffic light algorithm, level 2 tweet classification and picture processing techniques, and level 4 support services. The tweet classification and image-processing algorithms, as well as the suggested smart traffic signal method, were tested for accuracy and efficiency. Table3 summarizes the key benefits of deploying smart highways.

Table 3 Advantages of using smart roads in cities

Advantage	**Description**
Decrease congestion	Traffic congestion can be lessened by keeping an eye on vehicle flow and modifying traffic lights
Enhance safety	Other vehicles, pedestrians, and bikes can all be detected and avoided using computer vision. In the event of a collision or criminal activity, smart gadgets can alert first responders
Charging toll fees	Utilizing license plate recognition, e-tolling may automatically collect tolls for highway and bridge crossings
Detecting and avoiding problem areas	Analytics about crossings and roads with a high crash rate can be found on smart roads. Planners can use it to assess whether to implement mitigating measures like speed limit adjustments, signage, or signals

Advantage	Description
Monitoring road conditions	Cities can survey and fix paving conditions faster with smart roads
Reducing pollution	A lot of pollution is produced by idling engines. Smart road technology can assist to optimize traffic flow to reduce congestion

5.3 IoT for Smart Parking

One of the essential challenges of ITS is to discover a reliable solution to the car parking problem [70]. Indeed, in the major cities in the world, locating available parking places is consistently challenging for drivers.

It becomes difficult with the enormous increasing number of private car users. This trouble takes the attention of researchers and industries. They try to find a solution to minimize searching times for the driver. Finding a parking solution may also lessen traffic congestion and road accidents. Indeed, seeking available parking and traffic congestion can be alleviated if drivers are told ahead of time about the availability of parking spaces near their planned destination. Thus, the main goal of the smart parking system is to send drivers the available parking locations near their destination. Recently, a huge number of smart parking was created and implemented. In such a system, the parking should be embedded with specific sensors to survey free available parking locations. Further, a fast processing unit should be employed to acquire useful information from data gathered from various sources. Figure 7 shows the primary architecture of IoT-based smart parking systems[71].

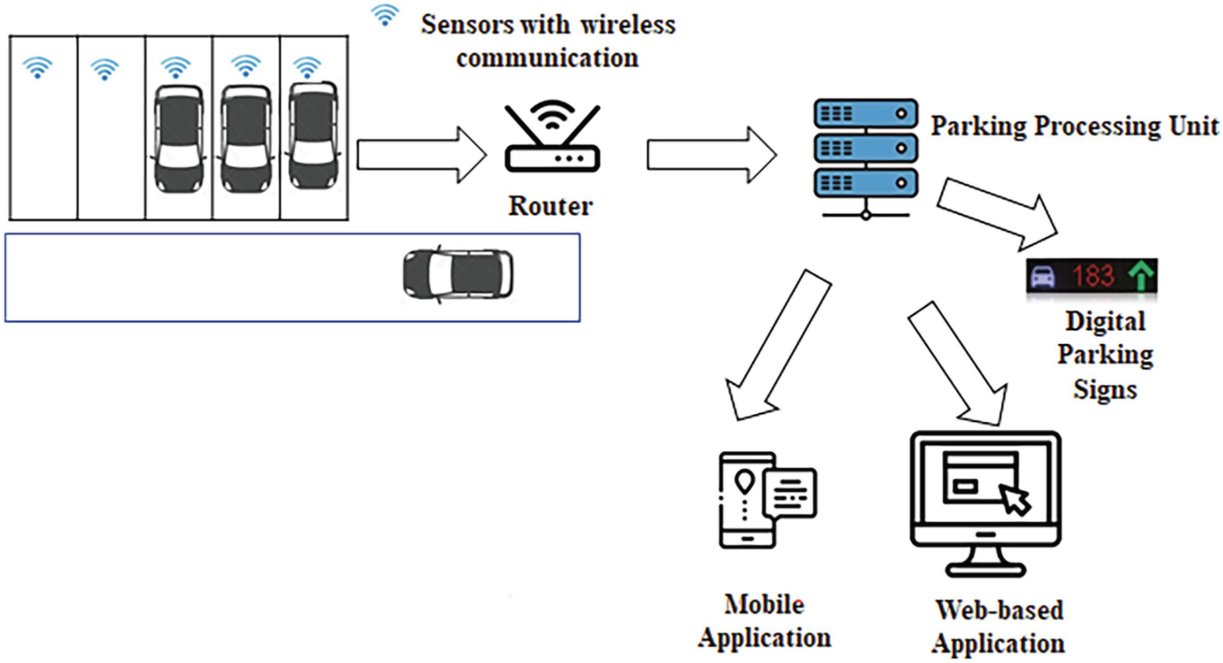

Sensors with wireless communication
Router
Parking Processing Unit
183
Digital Parking Signs
Mobile Application
Web-based Application

Fig. 7 General architecture of IoT-based smart parking

IoT-based smart parking system is composed of the following:

(a) Sensors, which indicate the availability of parking locations. These sensors should be embedded with wireless communication to send the data to the processing unit.

(b) A router, which facilitates the exchange of data between sensors: application, processing unit, and display screen.

(c) A processing unit, which examines gathered data and sends messages to the drivers via a web-based or mobile application.

(d) A parking screen, which receives information from the processing unit and displays the available parking locations accordingly.

(e) A parking application (a mobile-based or a web-based application), which receives essential information from the processing unit; it manages available parking spaces to redirect cars.

IoT and cloud computing have witnessed considerable improvement during the last decade. While each technology has its own benefits, their combination is desired to deliver meaningful advantages that complete one another. In fact, by using the boundless capabilities and resources of the cloud, the Internet of Things (IoT) can hold its technological constraints, such as those related to storage, processing, and power. Moreover, using IoT, the cloud can get outside its inability to interact with physical objects in a more distributed and dynamic way. Typically, the cloud serves as a go-between for devices and applications [72]. Below are some advantages of combining cloud computing and the Internet of Things [73]:

(a) Storage size: IoT consists of many objects that are connected to share data and valuable information. These objects may generate enormous amounts of unstructured or semi-structured data. IoT needs to collect, get access to, process, display, and share massive volumes of data. Utilizing the cloud is the best and most economical approach to manage the data generated by IoT. In fact, the cloud may provide limitless, inexpensive, and flexible storage. Users can access and visualize cloud-based data from any location using standard APIs.

(b) Processing power: There is not much computing power in the IoT devices that are in use. Therefore, smarter devices are used to aggregate and process data collected from numerous sensors. The computational needs of IoT can be handled by utilizing the cloud's infinite processing capacity and on-demand model. IoT devices could handle data in real time with the use of cloud computing. Thus, the application may be fast and very responsive.

(c) Communicative tools: The main functionality of IoT is to allow connected devices to communicate with each other through a specific set of hardware. Devices may be connected, tracked, and managed efficiently and from any location using the cloud platform. Using built-in apps, IoT systems can monitor and control items in real time from remote locations.

(d) Scalability: A scalable IoT strategy is provided by the cloud. It enables dynamic resource expansion and contraction. Any number of "things" could be added to or removed from the system if cloud integration is made available. The cloud allocates resources by the needs of devices and apps.

(e) Availability: by using cloud, locating resources becomes very simple at any time and from any location.

(f) Interoperability: The Internet of Things employs a wide range of devices. Compatibility issues may emerge as a result of these devices' differing hardware or software configurations. It becomes extremely difficult to guarantee device compatibility in an IoT ecosystem. The cloud can help to overcome this challenge since it provides a universal platform for several devices to connect and communicate. Devices can share and exchange data in ways that suit them.

Many recent smart parking systems are built based on IoT and cloud integration. In [74], the authors proposed a novel smart parking system using cloud computing and IoT. The WSN (wireless sensor network), which consists of RFID technology, is used by the proposed system to monitor parking lots. The percentage of open parking spaces in each parking lot is calculated using an RFID (radio frequency identification) reader. A large-scale system was implemented at a minimal cost, thanks to the usage of RFID. The system offers a way to settle disagreements in the parking lot and reduces time spent hunting for a place. The user can select an appropriate parking place after logging into the system. The user will receive confirmation about the chosen parking place through notification. The system then changes the parking space's status to "pending," at which point no other users will be able to reserve it.

Using the most recent long-range low-power wireless communication technologies, edge-cloud computing, and multi-parametric parking slot sensor nodes, the authors of [75] proposed a multi-layer design for a smart parking system. The suggested method provides dynamic management of parking for large regions while providing drivers with useful information about available parking spots and related services by monitoring cars in real time. The suggested system is capable of a dynamic pricing mechanism as well. It can provide the best parking availability for drivers and the maximum revenue for the parking authority.

6 Conclusion

Every year, traffic-related accidents result in thousands of fatalities, innumerable disabilities, and injuries, as well as environmental pollution that fuels additional tragedies. The economic and health effects of traffic congestion are most profoundly affected by delays in completing routine tasks, which lead to a major loss of time, effort, and focus as well as a variety of additional costs. As a result, developing a smart transportation system becomes crucial for cities and governments all over the world. In order to solve this problem, recent advancements in digital technology can be quite beneficial. As a result, IoT has transformed everything in our environment into intelligent objects with a unique identity, the ability to sense and gather information from their surroundings, share it via RFID and WSNs, and communicate with other intelligent objects.

In this chapter, the role of IoT in intelligent transportation systems is described. First, the IoT platform was defined and described. After that, the ITS was presented. Then, IoT role in ITS was presented. The authors focused on three main parts of ITS: smart traffic management system, smart road, and smart parking. The integration of IoT in Intelligent Transportation Systems (ITS) provides various advantages, such as environmental conservation through the reduction of waiting times and the time spent searching for parking spaces; increased company performance by minimizing transportation time; and increased safety by creating smart roads and improved vehicle communication and by providing a better customer experience, both for those waiting for goods and for those who use public transportation or electric and autonomous vehicles. Additional challenges of IoT integration in ITS, such as cost-benefit calculations, infrastructure security, and user data privacy, should be addressed in the future.

References

1. Kuddus, M.A., Tynan, E., McBryde, E.: Urbanization: a problem for the rich and the poor? Public Health Rev. **41**, 1–4 (2020)
2. Petrović, N., Bojović, N., Petrović, J.: Appraisal of urbanization and traffic on environmental quality. J. CO2 Utilization. **16**, 428–430 (2016)
3. Arena, F., Pau, G., Severino, A.: A review on IEEE 802.11 p for intelligent transportation systems. J. Sens. Actuat. Netw. **9**(2), 22 (2020)

4. Boukerche, A., Tao, Y., Sun, P.: Artificial intelligence-based vehicular traffic flow prediction methods for supporting intelligent transportation systems. Comput. Netw. **182**, 107484 (2020)
5. Chen, C., Liu, B., Wan, S., Qiao, P., Pei, Q.: An edge traffic flow detection scheme based on deep learning in an intelligent transportation system. IEEE Trans. Intell. Transp. Syst. **22**(3), 1840–1852 (2020)
6. Kosmatos, E.A., Tselikas, N.D., Boucouvalas, A.C.: Integrating RFIDs and smart objects into a unified internet of things architecture. Adv IoT: Scientif Res. **1**, 5–12 (2011)
7. Madakam, S., Lake, V., Lake, V., Lake, V.: Internet of things (IoT): A literature review. J. Comput. Commun. **3**(5), 164 (2015)
8. Nord, J.H., Koohang, A., Paliszkiewicz, J.: The internet of things: review and theoretical framework. Expert Syst. Appl. **133**, 97–108 (2019)
9. Nasserddine, G., Nassereddine, M., El Arid, A.A.: Internet of things integration in renewable energy Systems. In: Handbook of Research on Applications of AI, Digital Twin, and Internet of Things for Sustainable Development, pp. 159–185. IGI Global (2023)
10. Mahmud, S., Ahmed, S., Shikder, K.: A smart home automation and metering system using internet of things (IoT). In: 2019 International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), pp. 451–454. IEEE (2019)
11. Zhao, X., Askari, H., Chen, J.: Nanogenerators for smart cities in the era of 5G and internet of things. Joule. **5**(6), 1391–1431 (2021)
12. Muthuramalingam, S., Bharathi, A., Rakesh Kumar, S., Gayathri, N., Sathiyaraj, R., Balamurugan, B.: IoT based intelligent transportation system (IoT-ITS) for global perspective: A case study. IoT Big Data Anal. Smart Gen. **1**, 279–300 (2019)
13. Patel, P., Narmawala, Z., Thakkar, A.: A survey on intelligent transportation system using internet of things. Emerging Research in Computing, Information, Communication and Applications: ERCICA. **2018**(1), 231–240 (2019)
14. Rose, K., Eldridge, S., Chapin, L.: The internet of things: an overview. ISOC. **80**, 1–50 (2015)
15. Sestino, A., Prete, M.I., Piper, L., Guido, G.: Internet of things and big data as enablers for business digitalization strategies. Technovation. **98**, 102173 (2020)
16. Car, T., Stifanich, L.P., Šimunić, M.: Internet of things (IoT) in tourism and hospitality: opportunities and challenges. Tour. South East Eur. **5**, 163–175 (2019)
17. Kassab, W.A., Darabkh, K.A.: A–Z survey of Internet of Things: Architectures, protocols, applications, recent advances, future directions and recommendations. J. Netw. Comput. Appl. **163**, 102663 (2020)
18. Aleksandrovičs, V., Filičevs, E., Kampars, J.: Internet of things: structure, features and management. Inform. Technol. Manag. Sci. **19**(1), 78–84 (2016)
19. Li, S., Xu, L.D., Zhao, S.: The internet of things: a survey. Inf. Syst. Front. **17**, 243–259 (2015)

20. Sfar, A.R., Chtourou, Z., Challal, Y.: A systemic and cognitive vision for IoT security: A case study of military live simulation and security challenges. In: 2017 International Conference on Smart, Monitored and Controlled Cities (SM2C), pp. 101–105. IEEE (2017)
21. Singh, R.P., Javaid, M., Haleem, A., Suman, R.: Internet of things (IoT) applications to fight against COVID-19 pandemic. Diabetes Metab. Syndr. Clin. Res. Rev. **14**(4), 521–524 (2020)
22. Haseeb, K., Almogren, A., Islam, N., Ud Din, I., Jan, Z.: An energy-efficient and secure routing protocol for intrusion avoidance in IoT-based *WSN*. Energies. **12**(21), 4174 (2019)
23. INFSO, D: Internet of things in 2020: roadmap for the future. INFSO D. **4** (2008) https://docbox.etsi.org/erm/Open/CERP%2020080609-10/Internet-of-Things_in_2020_EC-EPoSS_Workshop_Report_2008_v1-1.pdf
24. Madakam, S., Ramaswamy, R., Tripathi, S.: Internet of things (IoT): A literature review. J. Comput. Commun. **3**, 164–173 (2015)
25. Dorsemaine, B., Gaulier, J.P., Wary, J.P., Kheir, N., Urien, P.: Internet of things: a definition & taxonomy. In: In 2015 9th International Conference on Next Generation Mobile Applications, Services and Technologies, pp. 72–77. IEEE (2015)
26. Jia, M., Komeily, A., Wang, Y., Srinivasan, R.S.: Adopting Internet of Things for the development of smart buildings: A review of enabling technologies and applications. Autom. Constr. **101**, 111–126 (2019)
27. Ramamurthy, A., Jain, P.: The Internet of Things in the Power Sector Opportunities in Asia and the Pacific. Asian Development Bank, Mandaluyong (2017)
28. Karunarathne, G.G.K.W.M.S.I.R., Kulawansa, K.A.D.T., Firdhous, M.F.M.: Wireless communication technologies in internet of things: a critical evaluation. In: 2018 International conference on intelligent and innovative computing applications (ICONIC), pp. 1–5. IEEE (2018)
29. Li, S., Da Xu, L., Zhao, S.: 5G internet of things: A survey. J. Ind. Inf. Integr. **10**, 1–9 (2018)
30. Gatsis, K., Pappas, G.J.: Wireless Control for the IOT: Power, Spectrum, and Security Challenges, pp. 341–342. Proceedings of the Second International Conference on Internet-of-Things Design and Implementation (2017)
31. Kaffash, S., Nguyen, A.T., Zhu, J.: Big data algorithms and applications in intelligent transportation system: A review and bibliometric analysis. Int. J. Prod. Econ. **231**, 107868 (2021)
32. Alam, M., Ferreira, J., Fonseca, J.: Introduction to Intelligent Transportation Systems, pp. 1–17. Intelligent Transportation Systems: Dependable Vehicular Communications for Improved Road Safety (2016)
33. Sattar, F., Karray, F., Kamel, M., Nassar, L., Golestan, K.: Recent advances on context-awareness and data/information fusion in ITS. Int. J. Intell. Transp. Syst. Res. **14**, 1–19 (2016)
34. Riaz, M.T., Aaqib, S.M., Ahmad, S., Amin, S., Ali, H., Husnain, S., Riaz, S.: The intelligent transportation systems with advanced technology of sensor and network. In: 2021 International

Conference on Computing, Electronic and Electrical Engineering (ICE Cube), pp. 1–6. IEEE (2021)

35. Al-Absi, M.A., Al-Absi, A.A., Lee, H.J.: Comparison between DSRC and other short range wireless communication technologies. In: 2020 22nd International Conference on Advanced Communication Technology (ICACT), pp. 1–5. IEEE (2020)

36. Kandris, D., Nakas, C., Vomvas, D., Koulouras, G.: Applications of wireless sensor networks: an up-to-date survey. Appl. Syst. Innov. **3**(1), 14 (2020)

37. Arena, F., Pau, G., Severino, A.: A review on IEEE 802.11 p for intelligent transportation systems. J. Sensor Actuat. Netw. **9**(2), 22 (2020)

38. Dimitrakopoulos, G.J., Uden, L., Varlamis, I.: The Future of Intelligent Transport Systems. Elsevier (2020)

39. Upadhyay, A., Kumar, A., Singh, V.: Traffic data collection and visualization using intelligent transport systems. In: Smart Cities—Opportunities and Challenges: Select Proceedings of ICSC, pp. 127–134. Springer, Singapore (2020)

40. Iskanderov, Y., Pautov, M.: Actor-network approach to self-organisation in global logistics networks. In: Intelligent Distributed Computing XIII, pp. 117–127. Springer International Publishing (2020)

41. Gallego-Madrid, J., Hermosilla, A., Skarmeta, A.F.: Dynamic deployment and testing of virtual on-board units in 5G. In: 2022 IEEE Future Networks World Forum (FNWF), pp. 305–309. IEEE (2022)

42. Croce, A.I., Musolino, G., Rindone, C., Vitetta, A.: Route and path choices of freight vehicles: A case study with floating car data. Sustain. For. **12**(20), 8557 (2020)

43. Li, Y., Zuo, X., Yang, F.: Research on urban resident activity patterns and hotspot area based on GPS floating car data. IEEE Access. **8**, 2694–2707 (2019)

44. Xu, X., Hu, R.: Dynamic traffic information estimation based on floating Car data. In: *Advances in Smart Vehicular Technology, Transportation, Communication and Applications: Proceeding of the Second International Conference on Smart Vehicular Technology, Transportation, Communication and Applications*, pp. 123–130. October 25–28; Mount Emei, China (2019)

45. Wochinger, K., Boehm-Davis, D.: Navigational preference and driver acceptance of advanced traveler information systems. In: *Ergonomics and Safety of Intelligent Driver Interfaces*, pp. 345–362. CRC Press (2020)

46. Jović, M., Tijan, E., Aksentijević, S., Sotošek, B., 2019. The Role of Electronic Transportation Management Systems in Seaport Digitalization

47. Saharan, S., Bawa, S., Kumar, N.: Dynamic pricing techniques for intelligent transportation system in smart cities: A systematic review. Comput. Commun. **150**, 603–625 (2020)

48. Chaudhary, R., Jindal, A., Aujla, G.S., Aggarwal, S., Kumar, N., Choo, K.K.R.: BEST: Blockchain-based secure energy trading in SDN-enabled intelligent transportation system. Comput. Secur. **85**, 288–299 (2019)

49. Hu, Q., Amini, M.R., Wiese, A., Seeds, J.B., Kolmanovsky, I., Sun, J.: A multirange vehicle speed prediction with application to model predictive control-based integrated power and thermal management of connected hybrid electric vehicles. J. Dyn. Syst. Meas. Control. **144**(1), 011105 (2022)

50. Saarika, P.S., Sandhya, K., Sudha, T.: Smart transportation system using IoT. In: The 2017 International Conference on Smart Technologies for Smart Nation (SmartTechCon), pp. 1104–1107. IEEE (2017)

51. Fong, S.L., Bakar, A.A.A.B.A., Ahmed, F.Y., Jamal, A.: Smart Transportation System Using RFID, pp. 579–584. Proceedings of the 2019 8th International Conference on Software and Computer Applications (2019)

52. Roy, A., Siddiquee, J., Datta, A., Poddar, P., Ganguly, G., Bhattacharjee, A.: Smart traffic & parking management using IoT. In: 2016 IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), pp. 1–3. IEEE (2016)

53. Godwin, J.J., Krishna, B.S., Rajeshwari, R., Sushmitha, P., Yamini, M.: IoT-based intelligent ambulance monitoring and traffic control system. Further Adv. IoT Biomed. Cyber Phys. Syst. **1**, 269–278 (2021)

54. Jan, B., Farman, H., Khan, M., Talha, M., Din, I.U.: Designing a smart transportation system: an internet of things and big data approach. IEEE Wirel. Commun. **26**(4), 73–79 (2019)

55. Bui, K.H.N., Jung, J.E., Camacho, D.: Game theoretic approach on real-time decision making for IoT-based traffic light control. Concurr. Comput. Pract. Exp. **29**(11), e4077 (2017)

56. George, A.M., George, V.I., George, M.A.: IoT based smart traffic light control system. In: The 2018 International Conference on Control, Power, Communication and Computing Technologies (ICCPCCT), pp. 148–151. IEEE (2018)

57. Abinaya, B., Gurupriya, S., Pooja, M.: Iot based smart and adaptive lighting in street lights. In: The 2017 2nd International Conference on Computing and Communications Technologies (ICCCT), pp. 195–198. IEEE (2017)

58. Badii, C., Bellini, P., Difino, A., Nesi, P.: Sii-mobility: an IoT/IoE architecture to enhance smart city mobility and transportation services. Sensors. **19**(1), 1 (2018)

59. Alvi, U., Khattak, M.A.K., Shabir, B., Malik, A.W., Muhammad, S.R.: A comprehensive study on IoT based accident detection systems for smart vehicles. IEEE Access. **8**, 122480–122497 (2020)

60. Shabir, B., Khan, M.A., Rahman, A.U., Malik, A.W., Wahid, A.: Congestion avoidance in vehicular networks: A contemporary survey. IEEE Access. **7**, 173196–173215 (2019)

61. Rahman, M.A., Ahsanuzzaman, S.M., Rahman, I., Ahmed, T., Ahsan, A.: IoT based smart helmet and accident identification system. In: 2020 IEEE Region 10 Symposium (TENSYMP), pp. 14–17. IEEE (2020)

62. Janani, R.P., Renuka, K., Aruna, A.: IoT in smart cities: A contemporary survey. Global Trans. Proc. **2**(2), 187–193 (2021)

63. Kiela, K., Barzdenas, V., Jurgo, M., Macaitis, V., Rafanavicius, J., Vasjanov, A., et al.: Review of V2X–IoT standards and frameworks for ITS applications. Appl. Sci. **10**(12), 4314 (2020)
64. Cohda Wireless. MK5 OBU Specification. Available online: http://www.neard.ws/wp-content/uploads/2018/11/MK5_OBU_Specification.pdf
65. Arada Systems. LocoMate OBU. Available online: http://www.aradasystems.com/wp-content/uploads/2015/06/Arada_datasheet_obu_2.01_2015.pdf
66. Arada Systems. LocoMate RSU. Available online: http://www.aradasystems.com/wp-content/uploads/2015/05/LocoMate_RSU_Datasheet_v2.0_May2015.pdf
67. Arada Systems. LocoMate ME, Mobile DSRC for V2X Connectivity. Available online: http://www.aradasystems.com/wp-content/uploads/2014/03/LocoMate_OBU_Datasheet_v2.16-MEREV1.16.pdf
68. Misra, S., Roy, C., Ghosh, R.: P-VERSE: prioritization of vehicles to enhance road safety in IoT environment. IEEE Internet Things J. (2023)
69. Alsaawy, Y., Alkhodre, A., Abi Sen, A., Alshanqiti, A., Bhat, W.A., Bahbouh, N.M.: A comprehensive and effective framework for traffic congestion problems based on the integration of IoT and data analytics. Appl. Sci. **12**(4), 2043 (2022)
70. Zhou, F., Li, Q.: Parking guidance system based on ZigBee and geomagnetic sensor technology. In: The 2014 13th International Symposium on Distributed Computing and Applications to Business, Engineering and Science, pp. 268–271. IEEE (2014)
71. Khanna, A., Anand, R.: IoT based smart parking system. In: The 2016 International Conference on Internet of Things and Applications (IOTA), pp. 266–270. IEEE (2016)
72. Lookmuang, R., Nambut, K., Usanavasin, S.: Smart parking using IoT technology. In: 2018 5th International Conference on Business and Industrial Research (ICBIR), pp. 1–6. IEEE (2018)
73. Sadeeq, M.M., Abdulkareem, N.M., Zeebaree, S.R., Ahmed, D.M., Sami, A.S., Zebari, R.R.: IoT and cloud computing issues, challenges and opportunities: A review. Qubahan Acad. J. **1**(2), 1–7 (2021)
74. Pham, T.N., Tsai, M.F., Nguyen, D.B., Dow, C.R., Deng, D.J.: A cloud-based smart-parking system based on Internet-of-Things technologies. IEEE Access. **3**, 1581–1591 (2015)
75. Sarker, V.K., Gia, T.N., Ben Dhaou, I., Westerlund, T.: Smart parking system with dynamic pricing, edge-cloud computing and lora. Sensors. **20**(17), 4669 (2020)
76. Zhou, X., Ke, R., Yang, H., Liu, C.: When intelligent transportation systems sensing meets edge computing: vision and challenges. Appl. Sci. **11**(20), 9680 (2021)
77. Liu, G., Shi, H., Kiani, A., Khreishah, A., Lee, J., Ansari, N., Yousef, M.M.: Smart traffic monitoring system using computer vision and edge computing. IEEE Trans. Intell. Transp. Syst. **23**(8), 12027–12038 (2021)

78. Chavhan, S., Gupta, D., Gochhayat, S.P.N.C.B., Khanna, A., Shankar, K., Rodrigues, J.J.: Edge computing AI-IoT integrated energy-efficient intelligent transportation system for smart cities. ACM Trans. Internet Technol. **22**(4), 1–18 (2022)

OceanofPDF.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024

S. Pal et al. (eds.), *IoT Edge Intelligence*, Internet of Things: Technology, Communications and Computing
https://doi.org/10.1007/978-3-031-58388-9_11

IoT-Driven Analytics and Edge Intelligence in Autonomous Navigation Systems

Wilson Nwankwo[1] ✉, Joy Orukwo[2], Chinecherem Umezuruike[3], Chukwuemeka Pascal Nwankwo[4] ✉, Paschal Uchenna Chinedu[5] ✉ and Chukwuemeka Obasi[6] ✉

(1) Department of Cyber Security, Delta State University of Science & Technology, Ozoro, Nigeria
(2) Department of Computer Science, Ignatius Ajuru University of Education, Port Harcourt, Nigeria
(3) College of Computing and Communication Studies, Bowen University, Iwo, Nigeria
(4) Department of Information Technology, Nigerian British University, Asa, Nigeria
(5) Department of Information Technology, Delta State University of Science and Technology, Ozoro, Nigeria
(6) Department of Computer Engineering, Edo State University, Uzairue, Nigeria

✉ **Wilson Nwankwo (Corresponding author)**
Email: wnwankwo@dsust.edu.ng

✉ **Chukwuemeka Pascal Nwankwo**
Email: nwankwo.pascal@nbu.edu.ng

✉ **Paschal Uchenna Chinedu**
Email: chinedupu@dsust.edu.ng

✉ **Chukwuemeka Obasi**
Email: chukwuemeka.obasi@edouniversity.edu.ng

Keywords IoT – Edge – Intelligence – Autonomous navigation system – Analytics – Integration

1 Introduction

The evolution and arrival of automated vehicles are due to exceptional progress in research that focused on IoT and embedded systems, edge computing, wireless communication, artificial intelligence, sensors, analysis and data acquisition, etc. A rapid and notable transformation in intelligent connectivity with regard to autonomous system is currently making recognizable waves in the world. Isolated systems have gradually given way to ubiquitous Internet-enabled “things” that provide the systems with capability to interact with each other, generate and process at the edge of the network, and offer control and regulation [1]. IoT and edge computing would change, exploit, and modify many aspects of the way we live [2]. The IoT evolution is driven by the predictability and forethought occasioned by the ability to follow objects by-way of the commodity chains against which they are rooted. A variety of connected technologies that work together constitute an IoT ecosystem. IoT devices that possess intelligence and interconnectivity may be stocked with implanted actuators, processors, sensors, and transceivers. The IoT actuators and sensors are devices that would aid interaction with the physical environment. The data collected by the sensors is intelligently stored and processed to provide subprime inferences and useful insights that may be remotely processed for immediate or delayed process control [3]. IoT, AI, and edge computing exhibit tremendous potentials as a trending reality especially as these technologies tend to complement each other; for instance, edge computing is thriving to usher in an interesting computing schemes including high-capacity analysis, control, and storage, closer to the edge of a network and thereby aiding the resolution of latency and scalability in systems. Thus, with such data processing capacity at the edge, the big data generated by connected IoT devices can be harnessed and processed and incremental learning may also be done at the edge, thereby evolving a progressively responsive and intelligent system. This lends credence to the characteristic offering of the needed approachability that would be useful toward the construction and deployment of autonomous navigation systems (ANS). The auspicious usage of autonomous navigation systems is quite appealing to humans as it does provide more conducive and effortless driving involvement in the near future.

Lately, IoT has been deemed very relevant in various domains such as autonomous vehicles [4], smart cities [5, 6], campus management [7], logistics [8], agriculture [9–16], and traffic management [12, 15]. Automated vehicles would be increasingly popular in public transportation systems as they would reduce congestion and improved traffic flow [17]. The successful adoption of autonomous vehicles will impact greatly on human lives by offering a more convenient and stress-free and safer driving experiences. In order to make autonomous vehicle safe, stable, established, and accurate, navigation data is pivotal [18]. Sensing the environment, connecting to the Internet, obeying traffic rules, self-navigation, parking, ensuring passenger and pedestrian safety, quick decision-making, etc., are attributes supported by autonomous vehicles [19].

Consequent upon the future development in the domain of autonomous navigation systems, this chapter provides a study of IoT-driven edge intelligence and its integration into the ANS space. The main object of this paper is to bridge the literature gap by providing comprehensive discourse on existing literature that covers the common practices and evolving technologies, challenges of IoT, edge computing, and unmanned vehicles.

2 Methodology

The authors adopted the Systematic Literature Review (SLR) approach. The SLR maps and assesses existing knowledge and identifies the gaps on specific issues toward producing further reliable knowledge. It aids the collection of related documents, journals, and publication that help provide extensive understanding of an ongoing study [20]. SLR’s three-pronged approach is shown in Fig. 1.

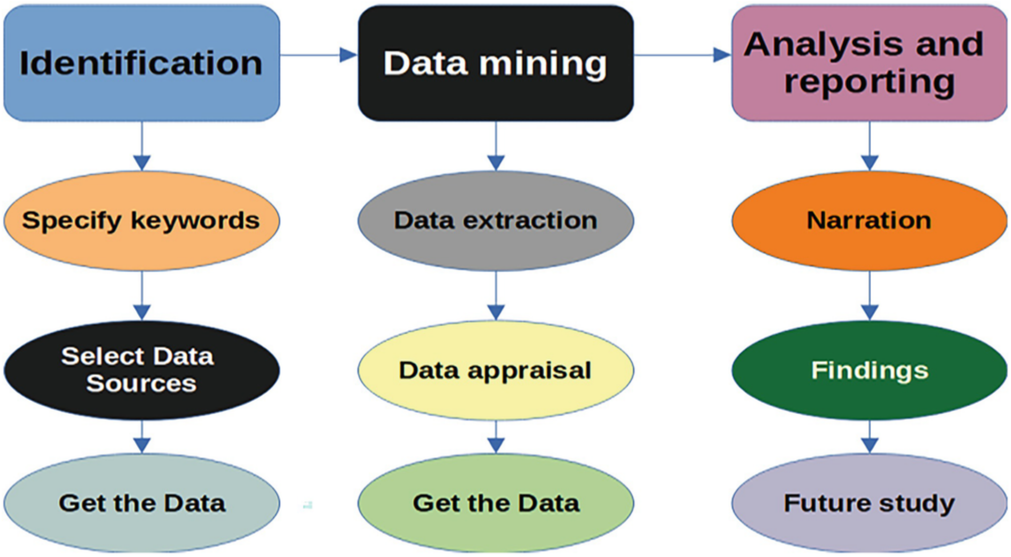

Identification
Data mining
Analysis and reporting
Specify keywords
Data extraction
Narration
Select Data Sources
Data appraisal
Findings
Get the Data
Get the Data
Future study

Fig. 1 SLR's three-pronged approach

2.1 Data Identification

This procedure includes three sub-steps: selection of keyword, selection of data source to be used, and selection of data for the study. Keywords selection is necessary as it aids the review in constructing search terms that are relevant and related to the topic. The keywords used are IoT, edge, edge intelligence, driverless car, self-driving cars, autonomous navigation system, automated system, and autonomous system. The data for this study was sourced from Scopus and EBSCO databases. In total, 180 papers were drawn, and duplicates removed leaving 120 papers.

2.2 Data Mining

Data extraction was done using the Preferred Reporting Items for Systematic Reviews and Meta-Analysis (PRISMA) approach as shown in Fig. 2. The title and abstract of each paper were examined to determine whether they included the keywords already stated. Furthermore, 72 papers that focus on IoT and edge intelligence in areas not relevant to autonomous navigation system were excluded leaving out 48 papers. The 48 papers included full research articles, reviews, and books. The predefined criteria for acceptance and rejection are shown in Table 1.

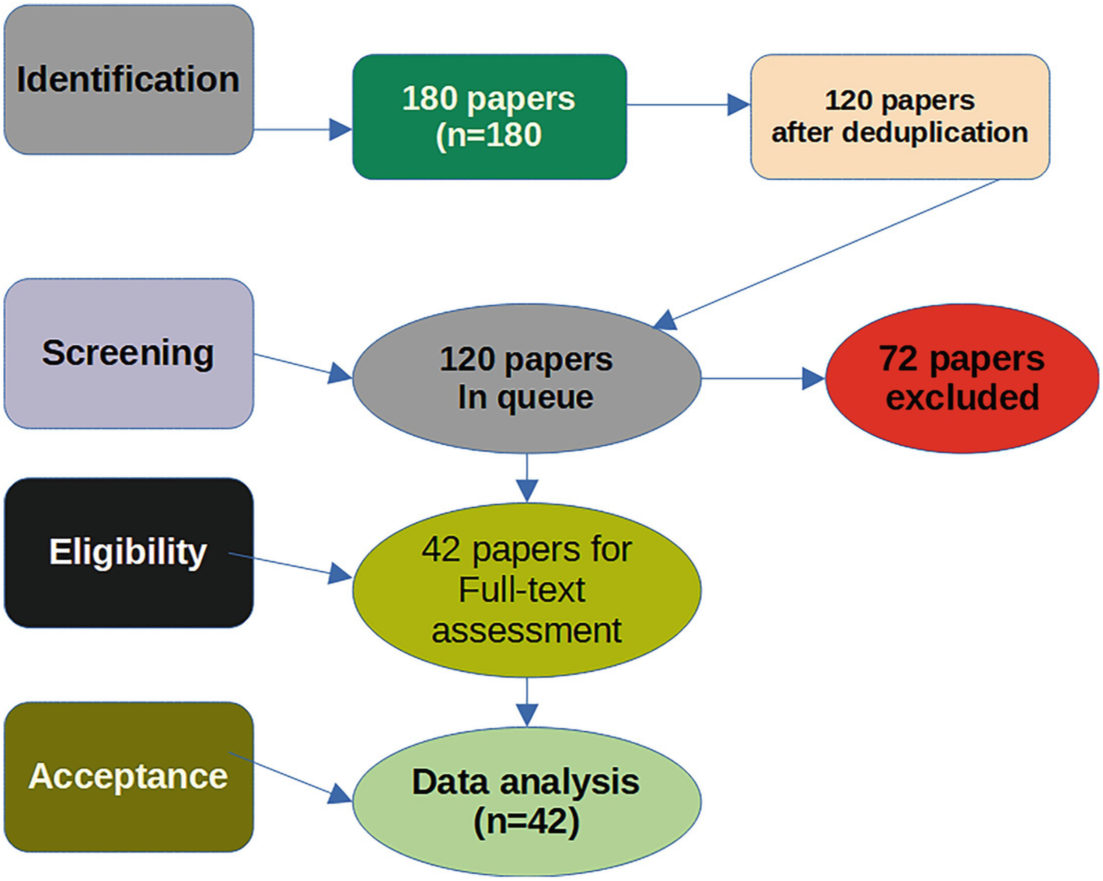

Identification
180 papers
(n=180
120 papers
after deduplication
Screening
120 papers
In queue
72 papers
excluded
Eligibility
42 papers for
Full-text
assessment
Acceptance
Data analysis
(n=42)

Fig. 2 The PRISMA workflow

Table 1 Predefined criteria for acceptance and rejection

S/N	Criteria	Decision
1.	Predefined keywords exist as a whole or at least in title, keywords, or abstract section of the paper	Accept
2.	Papers written in the English language	Accept
3.	Articles address IoT and automation	Accept
4.	Papers are published in a scientific and reputable journal	Accept
5.	Papers are duplicated in the search documents	Reject
6.	Papers are not accessible for review	Reject
7.	Papers published before 2016	Reject

2.3 Data Analysis and Result Reporting

Thematic and descriptive analysis were used. Thematic analysis provides comprehensive findings while descriptive analysis was used to help summarize data in a constructive way. Thematic analysis revealed that out of the 48 papers analyzed, 20 papers were most relevant for the study. Table 2 shows the results of the analysis.

Table 2 Thematic analysis of 20 most relevant papers

Author (s) and year	Paper title	Key contribution	Perceived limitations
[21]	Internet of vehicles: From intelligent grid to autonomous cars and vehicular fogs	Characteristics in emerging vehicles were discussed. Other issues like vehicular fog and its application to autonomous cars were outlined	No potential design was implemented for autonomous vehicles (AUV)
[22]	Results, issues and future challenges faced in autonomous cars	Categorized design and implementation problems into subunits such as software complexity, digital map construction, cost, simulation and validations while reviewing decision-making, privacy, resource computation, and privacy	Focused only on non-technical and social issues that concern autonomous driving
[23]	A scientometric and bibliometric review on autonomous vehicles	Study was based on progression, features, and trends as it relates to autonomous driving. They used a software called cite space to identify 96 fields	No data source was cited for the 96 identified fields
[24]	Aiding defense with autonomous vehicle control using IOT	Proposed a prototype automated vehicle using IoT for defense application with the aim of reducing manual drives. Existing methodology and setbacks were investigated	Failed to cover object detection and identification by autonomous vehicle to enhance its defense
[25]	An autonomous vehicle navigation system based on inertial and visual sensors	Parameters of line features from a camera was merged with Inertial Measurement Unit (IMU) data to improve the navigation accuracy. Their proposed system improves accuracy, stability, autonomy and reliability of navigation system and lessen SINS drift.	The paper only lays a theoretical foundation for Inertial Measurement Unit
[26]	A systematic literature review on capability, impact, planning, and policy: Understanding autonomous vehicles	They carried out a review of existing literature to acknowledge the impact, issues in planning and policy, and gave insight on possible bridge in the literature. They also advocated for the need	They used a manually handled literature review approach which doesn't make provision for the use of analytical techniques

Author (s) and year	Paper title	Key contribution	Perceived limitations
		of preparing for automated vehicles	
[27]	Recent advances, requirements, and challenges considering autonomous driving cars in smart cities	Their work cited advancement made in autonomous driving using fault tolerance, architecture, resource management, strict latency, and security as the key prerequisite for successful deployment of autonomous driving cars	No substantial literature on the subject matter was discussed
[28]	Reliability assessment of autonomous systems	Using PRISMA approach, they discussed the efficiency of autonomous systems and reviewed problems related to autonomous systems safety	However, they were unable to review components that aid the reduction of failure and enhance system reliability
[29]	IoT based self driving car	Their paper used raspberry pi and Arduino UNO to represent a mini version of self-driving car. They described a working method for autonomous vehicles.	Did not add multiple cameras and sensors to improve the accuracy of the mini self-driving car
[30]	Common practices and emerging technologies: a survey to autonomous driving	They presented a review of issues such as social impact, system architecture, object, and image detection. They weighed up datasets available for autonomous driving and the real-world setting.	However, their review algorithm lacked accuracy and efficiency
[31]	Autonomous remote security and mobile surveillance using Internet of Things: a review	Their paper entails idea about related devices and Internet of Things, its possibilities, and capabilities in providing high-level security	Their paper did not cover upcoming modules that would aid IoT advancement
[32]	Design of autonomous vehicle control using IoT	Their paper explained a prototype of an autonomous vehicle system using computer vision algorithm	However, they were unable to ensure computer communication security and maintenance
[33]	A survey of autonomous vehicles: Enabling communication technologies and challenges	They provided an extensive research on AV technologies, its advantages, disadvantages, and drawbacks. They surveyed 5G technology in the development of automated cars	However, they were unable to survey security and technical issues related to the deployment of the 5G-NR technology in vehicular communication

Author (s) and year	Paper title	Key contribution	Perceived limitations
[34]	Design and implemented a golf cart autonomous vehicle based on IoT	They described the development of a simulated autonomous golf-cart that can serve as a test-bed for other research projects, explained the conversion of golf-cart from the conventional way to autonomous way	However, improvements for the golf-cart autonomous system was only highlighted and no improvement was made
[35]	Automotive intelligence embedded in electric connected autonomous and shared vehicles technology for sustainable green mobility	These authors provided an overview in AI technologies advancement and applications for smart functions in future ECAS vehicles	However, they were unable to develop a new approach for vehicle operation system due to the complexity of the ECAS
[36]	Application of autonomous navigation in robotics	Various applications of autonomous navigation were discussed. Also, key technologies of autonomous navigation were introduced	However, the research was unable to explore renewal and iteration of crucial technologies and the maturity of evolving technologies, robots
[37]	Recent trends on IoT systems for traffic monitoring and for autonomous and connected vehicles	The authors focused on near real-time urban traffic analysis and prediction for effective intelligent transport systems using CNN for vehicle count generation	However, they were unable to analyze near real-time implementation at any CCTV location
[38]	Autonomous navigation technology for low-speed small unmanned vehicle: An overview	Their paper reviews technologies that are key of autonomous navigation. The authors analyzed autonomous navigation systems including their evolution in recent years	However, they were unable to review algorithms that improve autonomous navigation performance
[39]	IoT as a high degree of autonomy system	Their article discussed issues related to the use of recent IoT technology in agriculture. Methods and consequence of introducing IoT in the agricultural industry	However, there was no section for literature review in the paper making it difficult to know the authors sources of information and its validity
[40]	Autonomous vehicles progress, methods and challenges: a review	Noted the challenges of autonomous vehicles. Investigated the technological advancements in autonomous driving and recent	Did not include emerging and available technologies and their individual functions such

Author (s) and year	Paper title	Key contribution	Perceived limitations
		advances and case studies on autonomous driving	as perception, vehicle control, etc.

3 Internet of Things (IoT) and Edge Intelligence

IoT is commonly referred to as a network of physical objects including traditional non-computerized devices. IoT provides infrastructure for global communication between humans and machines. It is not only a network of computers but a network of devices of all type and sizes, vehicles, smart phones, home appliances, toys, cameras, medical instruments and industrial systems, animals, people, buildings, all connected, communicating, and sharing information based on stipulated protocols in order to achieve smart reorganizations, positioning, tracing, safety, control, real-time online monitoring, online upgrade, process control, and administration [41]. IoT advances the power and applicability of edge computing by enabling more connectedness and the generation of useful data which could be used to generate intelligent insights at the edge. Each major component in an IoT apparatus is a node. Examples of a node are sensors, controllers, actuators, humans, animals, storage server, edge computer, etc. [42]. Major applications of IoT and edge computing are smart city, smart industry, smart transport, smart buildings, smart energy, smart manufacturing, smart environment monitoring, smart living, smart health, smart food and water monitoring, etc. Various IoT sensors are embedded for sensing purposes such as temperature, humidity, pressure, etc. After sensing, these data are transmitted to a cloud server via an IoT gateway. Users and processing units could access the data through mobile apps, web apps, desktops apps, etc. IoT devices operate solitarily with their hearing and transmission [43]. An IoT arrangement includes four basic functional elements: data, device, analytics, and connectivity. Analytics encompass all workflows that enable the transformation of accumulated data into useful insights, whereas the connectivity ensures data sharing among participating nodes.

3.1 IoT Characteristics

Reference [44] enumerated the major characteristics of IoT as follows:

1. Security. This includes customer information and welfare security and this is achieved by generating a measurable safety model [43].

2. Dynamic changes: drastic change such as connected, disconnected, position, speed, etc., can happen to device and modify energetically.

3. Heterogeneity: numerous IoT nodes should connect to a variety of networks and hardware platforms.

4. Connectivity: IoT should present capabilities that aid accessibility and compatibility for content creation and usage [44].

5. Services related to things: IoT should present services related to things within things.

6. Inter-connectivity: IoT should be able to link to universal data and organizational and physical structures [45].

7. Enormous scale: The number of IoT nodes available should cover content semantics, with content management based on magnitude.

8. Safety: IoTs should guarantee customers/users security [45].

3.2 IoT Architecture

IoT system architecture consists of four layers: Object sensing layer, data exchange layer, information integration layer, and the application service layer.

1. Object Sensing Layer: This is the lowest layer. Smart objects integrated with sensors make up this layer. The sensors enable the interconnection of the physical and digital worlds allowing real-time information to be collected and processed. A variety of sensors are used for various purposes and are grouped according to their functions. These sensors have the ability to take measurements such as temperature, air quality, speed, humidity, pressure, flow, movement, and electricity. In some cases, they may also have a degree of memory, enabling them to record a certain number of measurements.

2. Data Exchange/Communication Layer: This layer ensures that communication networks transmit transparent data.

3. Information Integration/Management Service Layer: This layer takes care of business and process rule engines. It carries out processing of the uncertain information acquired from the networks by providing rule engines support for the formulation of decision logics and trigger interactive and automated processes to enable a responsive IOT system. In the area of analytics, various analytics tools are used to extract relevant information from massive amount of raw data and to be processed at a much faster rate [46].

4. Application Layer: This layer caters for domain intelligent environments/spaces such as transportation, building, city, lifestyle, retail, agriculture, factory, supply chain, emergency, health care, user interaction, culture and tourism, environment, energy, etc. [46]. Figure 2 shows an illustration of IoT architecture.

3.3 Challenges of IoT

IoT like other disruptive technologies is evolving. There are some critical issues associated with IoT. They are briefly presented herein.

1. Reliability: This is intended at increasing IoT service delivery success rate. It refers to proper working of the system based on its specification. Hence, reliability must be implemented throughout all IoT layers [41].

2. Performance: The quality of service (QoS) provided to the users depicts its overall performance. Performance evaluation of individual underlying protocols and technologies, application layer protocols, and QoS still remains an open issue in IoT.

3. Interoperability: Arising from the need to handle a large number of heterogeneous things that belong to different platforms, interoperability should be considered by both application developers and IoT device manufactures to ensure the delivery of services to all clients notwithstanding their hardware platform specifications [41].

4. Security and Privacy: Due to lack of common architecture and IoT security standard, it is not easy to guarantee user security and privacy. IoT is deeply rooted in the exchange of information between billions/trillions of connected internet objects.

5. Manufacturing: Manufacturing challenges such as cost and volume of production must be solved. The costs should be reduced; production process and products that are tagged sensor-enabled must have reduced impact on the environment.

3.4 IoT Analytics

Data in itself may not useful until it is analyzed, and patterns are identified which becomes a tool for decision-making. The analysis of data had become complex in recent times owing to the velocity, variety, and volume. Data generated by IoT devices falls into this category, and hence they are often called big data. Transforming data into useful information or knowledge for decision makers, designers, and developers of autonomous navigation system entails using technological tools relevant for prescription, description, diagnostic, and prediction [47]. Use of big data analytics tools to harness the data from IoT system may be called IoT analytics [48]. The tools applied would depend on the challenges and data available. IoT analytics is divided into four major types:

1. Prescriptive analytics: helps designers and developers to understand how to improve efficiency and output and prevent failures.

2. Descriptive analytics: involves detection of anomalies and monitoring of device performance to ensure that it is working according to specification.

3. Diagnostic analytics: understanding anomalies, inefficiencies, and their associated trend is the output of diagnostic analytics.

4. Predictive analytics: applying artificial intelligence to the vast amount of data generated by IoT devices to predict the future is what predictive analytics represent.

Machine learning (ML) has emerged as a popular tool for analyzing data generated by IoT devices. ML could be used to improve quality of service as well as for predictions in various domains [49–54]. Presently, the popular ML services offered to the public for harnessing IoT data include ML as a service (MLaaS), e.g., Microsoft Azure ML and Apache Mahout [55, 56]. There are also frameworks for IoT analytics. They include H2O.ai, Deep learning 4j, Tensorflow [57–60].

3.5 IoT and Edge Intelligence

Edge intelligence also called edge artificial intelligence (EAI) provides some remarkable enhancement to IoT devices making smarter. With the machine learning capabilities provided by EAI, edge computing devices which can constitute part of an IoT arrangement, can process complex and big data locally, incrementally learn from online data, as well as provide useful insights. For instance, an edge IoT device installed in a manufacturing site can track the conditions of specific machineries, thereby helping manufacturing facilities undertake effective predictive maintenance and preventing costly system failures and production losses. In autonomous systems, smart cameras with EAI chips could easily identify human and vehicular traffic and devise alternative routes. The benefits of EAI to IoT include the following:

(a) Reduced latency

(b) Reduced bandwidth requirement

(c) Flexibility

(d) Analytics

(e) Virtualization

EAI readily transforms IoT as it does make real-time data analysis and progressive intelligence possible. More so, EAI enhances operational efficiency by minimizing the data processing and storage costs.

4 Autonomous Navigation Systems

Autonomous navigation includes all computerized systems and machines that exhibit the capacity to navigate through pathways independently and intelligently. These systems are often equipped with IoT components and EAI. Sensor integration is a key critical functionality to the successful implementation of navigation systems. Autonomous systems play a vital role in assisting humans in a variety of problem areas. These include driverless cars, humanoid robots, assistive systems, domestic systems, military systems, and manipulator systems, etc. ANS provides optimal path from one location to another [61]. Traditional systems could be converted into automated systems by the addition of some components and sensors that provide information for decision-making by sensing the environment and controlling the mobility of the vehicle [62]. This is the realm of analytics. The implication is that data gathered via IoT components such as sensors must be implicitly analyzed and used in the navigation process. Figure 3 shows an automated system architecture. The model comprises four stages or layers namely perception, decision and planning, control, and chassis. Generally speaking, the technology in autonomous navigation mainly involves environment awareness technology, map building technology, motion planning technology, and tracking control technology, and all these lean on intelligent analytics [63].

1. Perception: AVs perceive their surrounding and detect the position in the environment via various sensors such as RADAR, LIDAR, camera, and real-time kinetic (RTK). The information collated from the sensor is passed to the recognition module, processed, fused, and passed to the decision and planning phase [64].

2. Decision and Planning: data collected from the perception stage is utilized here in planning and controlling movement and behavior of AV to aid decision-making using current and past information available on real-time map information, traffic details and patterns, information by the user, etc., to plan path, avoid obstacle, predict action, etc. Errors and information are recorded for future reference [65].

3. Control: receives data from the decision and planning phase and carries out functions that relates to AVs physical control such as steering,

braking, accelerating, etc. [33].

4. Chassis: this last phase includes the interface with the mechanical components mounted on the chassis such as the accelerator pedal motor, brake pedal motor, steering wheel motor, and gear motor. The control module controls and signals these components.

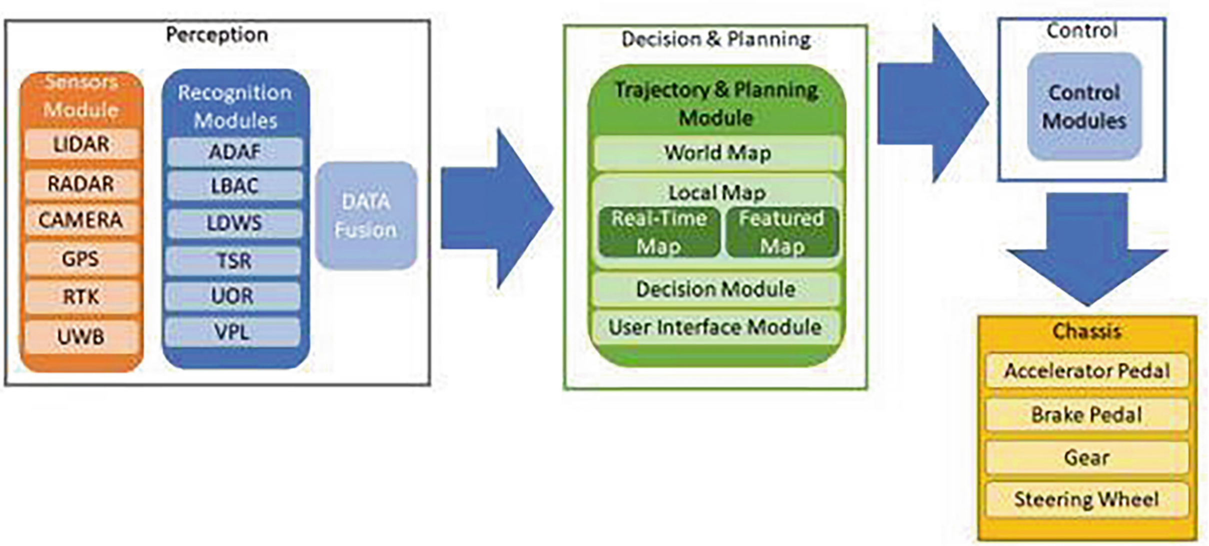

Perception
Sensors Module
LIDAR
RADAR
CAMERA
GPS
RTK
UWB
Recognition Modules
ADAF
LBAC
LDWS
TSR
UOR
VPL
DATA Fusion
Decision & Planning
Trajectory & Planning Module
World Map
Local Map
Real-Time Map
Featured Map
Decision Module
User Interface Module
Control
Control Modules
Chassis
Accelerator Pedal
Brake Pedal
Gear
Steering Wheel

Fig. 3 Automated system architecture

5 IoT and Edge Intelligence (EAI) Integration in Autonomous Navigation System

IoT and EAI could enable autonomous systems in different ways and enhance these functions in different use cases. The combined autonomous vehicles and IoT-EAI ecosystems implemented in the AUTOPILOT use cases integrate the services provided by harmonious IoT platforms and IoT devices that provide additional information to the vehicles about the environment, surroundings, and the dynamic events around the vehicles to enhance its automated functions [66]. Driverless cars are a perfect example of IoT-EAI integrated in autonomous systems. These cars contain several sensors, actuators, and micro controllers. Just like every computer system, there is an input (sensor), output, and a processor (microcontroller). In the system, the microcontroller is the processor, and it processes what comes in from the sensors (the sensors collate surrounding data continuously and sends it to the cloud/central unit) and gives out an output signal to actuators; for example, if we want to head to certain destination in a driverless car, we will input our destination either by text or by speech. If we choose to use the text option, then we will have to type our destination using a keypad, and if we go by voice, then we will use a condenser mic (mouth piece). Whichever option we choose, the microcontroller receives it as an input, converts the input signal to the corresponding coordinates (longitude and latitude), and sends it to the GPS module usually connected to a network (Wi-Fi, SIM, module, etc.) for Internet access. Once the GPS module is connected to the Internet, it sends the coordinates to the cloud where it can access a map (Google map). Once the map has been accessed, the car begins to accelerate and other sensors are activated. If, for any reason, a car stopped at the middle of the road, the accelerometer (sensor) detects it and the proximity sensor measures the distance of the car that broke down to it. Once the distance is known, the microcontroller sends a signal to the motor (actuator), which will cause the car to change direction, to the left or right as the case maybe. As the user keeps using the car, the car learns and makes decisions like best route to a destination, default address, favorite spot, etc.

6 Conclusion

The driving force behind the fusion of edge artificial intelligence (EAI) and IoT lies in the need for intelligent, real-time decision-making at the edge of the network. Several key factors that contribute to this trend include reduced latency, bandwidth optimization, enhanced privacy and security, improved reliability, scalability and flexibility, and cost-efficiency. Underlying any useful IoT implementation in a real-life situation is intelligent analytics. Autonomous navigation system is a trending development that exploits the capabilities of IoT and EAI as well as analytics. IoT component alone does not ensure the approachability needed to deploy autonomous navigation systems. In this chapter, we have adopted the SLR approach to conduct a study on the present development of IoT-driven analytics and edge intelligence in autonomous navigation systems. 42 papers were selected and reviewed based on pre-defined criteria. From our findings, ANS, IoT, EAI, and analytics are all evolving technologies and exhibit great potentials in the future technology ecosystem. We conclude that future research in these areas would enhance the applicability and operationalization of autonomous navigation systems, which will usher in the next evolutionary phase of the Internet of vehicles.

---

7 Future Direction

As we navigate the landscape of IoT-driven analytics and edge intelligence in autonomous systems, several promising avenues for future research emerge. These directions aim to further enhance the understanding, application, and impact of these technologies in the field of autonomous navigation. Some of the notable future engagements are discussed briefly.

First, the introduction of 5G into the scene will necessitate the need to investigate the implications of IoT-driven analytics and edge intelligence with 5G and future communication technologies. For instance, it will be interesting to establish how higher data transfer speeds and reduced latency can further optimize decision-making in autonomous navigation systems.

Second, we will examine the potential of leveraging IoT and edge intelligence to enable autonomous swarm systems. Here, we will investigate how these technologies can enhance coordination and communication among a network of autonomous entities, leading to more efficient and collaborative navigation. Exploring the role of IoT-driven analytics and edge intelligence in facilitating effective collaboration between autonomous systems and human operators. In the above subject, efforts will be directed toward investigating user interfaces, decision support systems, and communication protocols to enhance human-autonomy synergy.

Third, autonomous systems, like humans, may be affected by adverse conditions; hence, it is important to address the challenges associated with adverse environmental conditions, such as extreme weather or unexpected obstacles. Research should focus on enhancing the robustness of autonomous navigation systems through improved sensor technologies and resilient decision-making algorithms.

Energy-efficiency is another issue that should be given attention. It is pertinent to note that optimization of energy efficiency in autonomous navigation systems will help in reducing the cost of operation. IoT-driven analytics and edge intelligence can contribute to intelligent energy management, extending the operational life of autonomous devices.

Beyond, traditional transportation and traffic management, ANS may offer benefits in other sectors. Therefore, investigating how these technologies can be applied to health care, agriculture, and smart cities, will broaden the impact of IoT and edge intelligence.

Finally, the exploration of the user-centric design principles in the development of autonomous navigation systems may offer immense benefits; hence, we think that future efforts may be directed at investigating how human preferences, perceptions, and interactions can be seamlessly integrated into the design process to foster greater acceptance and usability.

References

1. Manoj, D., Kanika, S., Ravi, D., Sheetal, B., Kapil, J., Prabhishek, S.: A review on autonomous remote security and mobile surveillance using internet of things. J. Phys. Conf. Series. **1854**, 1–5 (2021)
2. Kumar, N., Agrawal, A., Khan, R.: METHWORK: An approach for ranking in research trends with a case study for IoET. Recent Adv. Comput. Sci. Commun. (Formerly Recent Patents on Computer Science). **4**(1), 7–11 (2019)
3. Yusuf, P., Mahmoud, A., Bedine, K., Hani, A.: An extended review on Internet of Things (IoT) and its promising applications. Commun. Appl. Electron. (CAE): Found. Comput. Sci. (FCS). **7**(26), 1–16 (2019)
4. Quy, V., Nam, V., Linh, D., Ban, N., Han, N.: Communication solutions for vehicle ad-hoc network in smart cities environment: a comprehensive survey. Wirel. Pers. Commun. **122**, 2791–2815 (2022)
5. Li, W., Logenthiran, T., Phan, V., Woo, W.A.: Novel smart energy theft system (SETS) for IoT-based smart home. IEEE IoT J. **6**, 5531–5539 (2019)
6. An, J., Le-Gall, F., Kim, J., Yun, J., Hwang, J., Bauer, M., Zhao, M., Song, J.: Toward global IoT-enabled smart cities interworking using adaptive semantic adapter. IEEE IoT J. **6**, 5753–5765 (2019)
7. Chang, Y., Lai, Y.: Campus edge computing network based on IoT street lighting nodes. IEEE Syst. J. **14**, 164–171 (2020)
8. Humayun, M., Jhanjhi, N., Hamid, B., Ahmed, G.: Emerging smart logistics and transportation using IoT and blockchain. IEEE Internet Things Mag. **3**, 58–62 (2020)
9. Adetunji, C.O., Osikemekha, A.A., Olaniyan, T.O., Daniel, I.H., Nwankwo, W., Olayinka, A.S.: Toward the design of an intelligent system for enhancing salt water shrimp production using fuzzy logic. In: Abraham, A., Dash, S., Rodrigues, J.J.P.C., Acharya, B., Pani, S.K. (eds.) Intelligent Data-Centric Systems: AI, Edge and IoT-based Smart Agriculture, pp. 533–541. Academic Press (2021a). https://doi.org/10.1016/B978-0-12-823694-9.00005-0

[Crossref]

10. Osikemekha, A.A., Adetunji, C.O., Olaniyan, T.O., Daniel, I.H., Nwankwo, W., Olayinka, A.S.: IoT-based monitoring system for freshwater fish farming: analysis and design. In: Abraham, A., Dash, S., Rodrigues, J.J.P.C., Acharya, B., Pani, S.K. (eds.) Intelligent Data-Centric Systems: AI, Edge and IoT-based Smart Agriculture, pp. 505–515. Academic Press (n.d.)
11. Olayinka, A.S., Adetunji, C.O., Nwankwo, W., Olugbemi, O.T., Olayinka, T.C.: A study on the application of bayesian learning and decision trees IoT-enabled system in postharvest storage. In: Pal, S., De, D., Buyya, R. (eds.) Artificial Intelligence-based Internet of Things Systems. Internet of Things (Technology, Communications and Computing). Springer, Cham (2022). https://doi.org/10.1007/978-3-030-87059-1_18
[Crossref]
12. Nwankwo, W., Adetunji, C.O., Olayinka, A.S.: IoT-driven Bayesian learning: a case study of reducing road accidents of commercial vehicles on highways. In: Pal, S., De, D., Buyya, R. (eds.) Artificial Intelligence-Based Internet of Things Systems. Internet of Things (Technology, Communications and Computing). Springer, Cham (2022). https://doi.org/10.1007/978-3-030-87059-1_15
[Crossref]
13. Nwankwo, W., Ukhurebor, K.E.: Big Data Analytics: A Single Window IoT-enabled Climate Variability System for all-year-round Vegetable Cultivation, pp. 655–12030. In 2021 IOP Conference Series: Earth and Environmental Science (2021). https://doi.org/10.1088/1755-1315/655/1/012030
[Crossref]
14. Nwankwo, W., Adetunji, C., Ukhurebor, K.E., Makinde, S., Ubi, B. The precursory machinery of internet of things (IoT) in the platform for harmonizing bio-mined data. Niger. Res. J. Eng. Environ. Sci., 2020. 5(2), 786–796. http://rjees.com/download/?file=V05-N02-786-796.pdf
15. Nwankwo, W., Olayinka, A.S., Ukhurebor, K.E.: The urban traffic congestion problem in Benin city and the search for an ICT-improved solution. Int. J. Sci. Technol. **8**(12), 65–72 (2019)
16. Nwankwo, W., Olayinka, A.S., Umezuruike, C.: Boosting self-sufficiency in maize crop production in Osisioma Ngwa local government with IOT-climate messaging: a model. Afr. J. Agric. Res. **14**(7), 406–416 (2019). https://doi.org/10.5897/AJAR2018.13753
[Crossref]
17. Parent, M.: Advanced urban transport: automation is on the way. Int. Underw. Syst. Des. **22**, 9–11 (2017)
18. Zhao, J., Liang, B., Chen, Q.: The key technology toward the self-driving car. Int. J. Intell. Autom. Syst. **6**, 2–20 (2018)
19. Zhang, T., Zhang, L., Zhao, L., Huang, X., Hou, Y.: Catalytic effects in the cathode of Li-S batteries: accelerating polysulfides redox conversion. Energy Chem. **2**, 100036 (2020)
20. Wondimagegn, M., Teshome, S., Gudina, L.: Method for conducting systematic literature review and meta-analysis for environmental science research. Sci. Total Environ. **7**(2), 100777–134581 (2020)

21. Lee, E., Gerla, M., Pau, G., Lee, U., Lim, J.: Internet of vehicles: from intelligent grid to autonomous cars and vehicular fogs. Int. J. Distrib. Sens. Netw. **12**(9), 1–14 (2016)

22. Hussain, R., Zeadally, S.: Autonomous cars: research results, issues, and future challenges. IEEE Commun. Surviv. Tutor. **21**, 1275–1313 (2018)

23. Gandia, R., Antonialli, F., Cavazza, B., Neto, A., De-Lima, D., Sugano, J., Nicolai, I., Zambalde, A.: Autonomous vehicles: scientometric and bibliometric review. Transp. Rev. **39**, 9–28 (2018)

24. Bharath, S., Marimuthu, K., Syed, A.: Aiding defense with autonomous vehicle control using IOT. Global J. Pure Appl. Math. **14**(3), 433–441 (2018)

25. Xingxing, G., Yanbin, G., Henry, L., Pan, L., Guangchun, L.: An autonomous vehicle navigation system based on inertial and visual sensors. Sensors. **18**, 1–17 (2018)

26. Faisal, A., Kamruzzaman, M., Yigitcanlar, T., Currie, G.: Understanding autonomous vehicles: a systematic literature review on capability, impact, planning and policy. J. Transp. Land Use. **12**, 45–72 (2019). https://doi.org/10.5198/jtlu.2019.1405.
[Crossref]

27. Yaqoob, I., Khan, L., Kazmi, S., Imran, M., Guizani, N., Hong, C.: Autonomous driving cars in smart cities: recent advances, requirements, and challenges. IEEE Netw. **34**, 174–181 (2019)

28. Kalesanwo, O., Awodele, O., Eze, M., Kuyoro, S.: Reliability assessment of autonomous systems: a systematic review. Int. J. Comput. Trend. Technol. **68**(3), 48–52 (2020)

29. Rahul, P., Abhishek, S., Zeeshaan, W., Vaibhav, U., Nasiruddin, M.: IoT based self driving car. Int. Res. J. Eng. Technol. (IRJET). **7**(3), 5177–5181 (2020)

30. Yurtsever, E., Lambert, J., Carballo, A., Takeda, K.: A survey of autonomous driving: common practices and emerging technologies. IEEE Access. **8**, 58443–58469 (2020)

31. Manoj, D., Kanika, S., Ravi, D., Sheetal, B., Kapil, J., Prabhishek, S.: A review on autonomous remote security and Mobile surveillance using internet of things. J. Phys.: Conf. Series, 1848–1854 (2020)

32. Manjunathan, A., Suresh, A., Udhayanan, S., Thirumarai, C., Alexander, A.: Design of Autonomous Vehicle Control Using IoT, pp. 1–5. IOP Conference Series: Materials Science and Engineering (2021). https://doi.org/10.1088/1757-899X/1055/1/012008
[Crossref]

33. Ahangar, M., Ahmed, Q., Khan, F., Hafeez, M.: A survey of autonomous vehicles: enabling communication technologies and challenges. Sensors. **21**, 706–719 (2021). https://doi.org/10.3390/s21030706
[Crossref]

34. Sivakumar, B., Nisarga, T.: Design and implementation of IoT based golf cart autonomous vehicles. Int. Res. J. Eng. Technol. (IRJET). **8**(9), 2081–2096 (2021)

35. Vermesan, O., John, R., Pype, P., Daalderop, G., Kriegel, K., Mitic, G., Lorentz, V., Bahr, R., Sand, H., Bockrath, S., Waldhör, S.: Automotive intelligence embedded in electric connected

autonomous and shared vehicles technology for sustainable green mobility. Front. Future Transp. **2**, 688482 (2021). https://doi.org/10.3389/ffutr.2021.688482
[Crossref]

36. Wenshuai, J.: Application of autonomous navigation in robotics. J. Phys.: Conf. Series. **19**(6), 1–11 (2021)

37. Sergio, S., Stefano, G., Riccardo, M.: Recent trends on IoT systems for traffic monitoring and for autonomous and connected vehicles. Sensors. **21**(5), 1648–1653 (2021). https://doi.org/10.3390/s21051648
[Crossref]

38. Li, X., Li, Q., Yin, C., Zhang, J.: Autonomous navigation technology for low-speed small unmanned vehicle: an overview. World Electr. Veh. J. **13**, 165 (2022)

39. Zyukin, D., Khasambiev, I.: IoT as a high degree of autonomy system. J. Phys. Conf. Ser. **2176**, 1–7 (2022). https://doi.org/10.1088/1742-6596/2176/1/012020
[Crossref]

40. Parekh, D., Poddar, N., Rajpurkar, A., Chahal, M., Kumar, N., Joshi, G., Cho, W.: A review on autonomous vehicles: Progress, methods and challenges. Electronics. **11**, 1–18 (2022)

41. Jordi, S., Santiago, S.: Internet of Things, 1st edn. Tech Pedia: Czech (2017)

42. Laith, F., Rupak, K., Omprakash, K., Marcela, Q., Ali, A.: A Concise Review on Internet of Things (IoT)—Problems, Challenges and Opportunities. 11th International Symposium on Communication Systems, Networks, and Digital Signal Processing (CSNDSP) (2018)

43. Nia, A., Jha, N.: A comprehensive study of the security of Internet-of-Things. IEEE Trans. Emerg. Top. Comput. **5**, 586–602 (2017)

44. Lakshmana, K., Selfedine, K.: Blockchain in the industrial Internet of Things. IOP Sci., 1–16 (2021)

45. Yang, Y., Wu, L., Yin, G., Li, L., Zhao, H.: A survey on security and privacy issues in internet-of-things. IEEE Internet Things J. **4**, 1250–1258 (2017)

46. Keyur, K., Sunil, M.: Internet of things-IOT: definition, characteristics, architecture, enabling technologies, application & future challenges. Int. J. Eng. Sci. Comput. **6**(5), 6121–6131 (2016)

47. DeTore, C.: IoT Analytics Applications and Use Cases. Retrieved from PTC (2021) https://www.ptc.com/en/blogs/iiot/iot-analytics-applications-and-use-cases

48. TIBCO, 2023. What is IoT Analytics? Retrieved from https://www.tibco.com/reference-center/what-is-iotanalytics#:~:text=Internet%20of\allowbreak{}%20\allowbreak{}Things%20(IoT)%20analytics,\allowbreak{}with%20Industrial\allowbreak{}%20IoT\allowbreak{}%20(IIoT)

49. Ravesa, A., Shabir, A.S.: Precision agriculture using IoT data analytics and machine learning. J. King Saud Univ.—Comput. Inf. Sci. **34**, 5602 (2022). https://doi.org/10.1016/j.jksuci.2021.05.013
[Crossref]

50. Nwankwo, W., Nwankwo, C.P., Adigwe, W.: Leveraging on artificial intelligence to accelerate sustainable bioeconomy. IUP J. Knowl. Manag. **20**(2), 38–59 (2022)

51. Chinedu, P.U., Nwankwo, W., Masajuwa, F.U., Imoisi, S.: Cybercrime detection and prevention efforts in the last decade: An overview of the possibilities of machine learning models. Rev. Int. Geogr. Educ. (RIGEO). **11**(7), 956–974 (2021). https://doi.org/10.48047/rigeo.11.07 [Crossref]

52. Acheme, D.I., Makinde, A.S., Osemengbe, U., Nwankwo, W.: An intelligent agent-based stock market decision support system using fuzzy logic. IUP J. Inf. Technol. **16**(4), 1–20 (2020)

53. Nwankwo, W., Ukhurebor, K.E., Ukaoha, K.C.: Knowledge discovery and analytics in process re-engineering: a study of port clearance processes. In: 2020 International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS), pp. 1–7. IEEE Explore (2020). https://doi.org/10.1109/ICMCECS47690.2020.246989 [Crossref]

54. Adetunji, C.O., Nwankwo, W., Olayinka, A.S., Olaniyan, O.T., Akram, M., Laila, U., Olugbenga, M.S., Oshinjo, A.M., Adetunji, J.B., Okotie, G.E., Esiobu, N. Machine Learning and Behaviour Modification for COVID-19. In: Hajiya Mairo Inuwa, Ifeoma Maureen Ezeonu, Charles Oluwaseun Adetunji, Emmanuel Olufemi Ekundayo, Abubakar Gidado, Abdulrazak B. Ibrahim, Benjamin Ewa Ubi(eds). Medical Biotechnology, Biopharmaceutics, Forensic Science and Bioinformatics, 2022. 271–287. Taylor & Francis. https://doi.org/10.1201/9781003178903

55. Kahveci, S., Alkan, B., Ahmad, M., Ahmad, B., Harrison, R.: An end-to-end big data analytics platform for IoT-enabled smart factories: a case study of battery module assembly system for electric vehicles. J. Manuf. Syst. **63**, 214–223 (2022). https://doi.org/10.1016/j.jmsy.2022.03.010 [Crossref]

56. Liu, Q., Chen, L., Jiang, H., Wu, J., Wang, T., Peng, T., Wang, G.: A collaborative deep learning microservice for backdoor defenses in industrial IoT networks. Ad Hoc Netw. **2022**, 124 (2021). https://doi.org/10.1016/j.adhoc,102727 [Crossref]

57. Jinya, S., Xiaoyong, Z., Shihua, L., Wen-Hua, C.: AI meets UAVs: a survey on AI empowered UAV perception systems for precision agriculture. Neurocomputing. **518**, 242–270 (2023)

58. Lang, M., Martin, B., Jakob, R., Patrick, S., Florian, P., Stefan, C., Quay, A., Giuseppe, C., Lars, K., Bernd, B.: MLR3: a modern object-oriented machine learning framework in R. J. Open Source Softw. **4**(44), 1903 (2019). https://doi.org/10.21105/joss.01903 [Crossref]

59. Varshini, H., Yogesh, S., Pasha, D., Suhail, M., Madhumitha, V., Sasi, A.: IoT-enabled smart doors for monitoring body temperature and face mask. Global Trans. Proc. **2**, 246–254 (2021). https://doi.org/10.1016/j.gltp.2021.08.071 [Crossref]

60. Nagaraju, R., Pentang, J., Abdufattokhov, S., Fernando, C., Mageswari, N., Uganya, G.: Attack prevention in IoT through hybrid optimization mechanism and deep. Meas. Sensors. **24**, 100431 (2022). https://doi.org/10.1016/j.measen.2022.100431 [Crossref]

61. Kolar, P., Benavidez, P., Jamshidi, M.: Survey of data-fusion techniques for laser and vision based sensor integration for autonomous navigation. Sensors. **20**(4), 2180 (2020)
62. Zanchin, B., Adamshuk, R., Santos, M., Collazos, K.: On the Instrumentation and Classification of Autonomous Cars, pp. 2631–2636. In Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC) (2017)
63. Wang, B., Han, Y., Tian, D., Guan, T.: Sensor-based environmental perception Technology for Intelligent Vehicles. J Sens. **8199361**, 1 (2021)
64. Llorca, F., Daza, G., Parra, N., Alonso, I.: Sensors and sensing for intelligent vehicles. Sensors. **20**, 511 (2020)
65. Zong, W., Zhang, C., Wang, Z., Zhu, J., Chen, Q.: Architecture design and implementation of an autonomous vehicle. IEEE Access. **6**, 21956–21970 (2018)
66. Ovidiu, V., Roy, B., Mariano, F., Daniel, B., Ilaria, B., Anton, D., Alexander, V., Mahdi, B.: IoT technologies for connected and automated driving applications. Int. Res. J. Eng. Technol. **8**(10), 259–306 (2018)

OceanofPDF.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024

S. Pal et al. (eds.), *IoT Edge Intelligence*, Internet of Things: Technology, Communications and Computing
https://doi.org/10.1007/978-3-031-58388-9_12

Edge-AI for Monitoring Air Pollution from Urban Waste Incineration: A Survey

Fatou Ciré Emilie Sané[1] ✉, Maissa Mbaye[1] ✉ and Bamba Gueye[2] ✉
(1) Université Gaston Berger, Saint-Louis, Senegal
(2) Université Cheikh Anta Diop, Dakar, Senegal

✉ **Fatou Ciré Emilie Sané (Corresponding author)**
Email: sane.fatou-cire-emilie@ugb.edu.sn

✉ **Maissa Mbaye**
Email: maissa.mbaye@ugb.edu.sn

✉ **Bamba Gueye**
Email: bamba.gueye@ucad.edu.sn

Keywords Waste incineration pollution – Edge-AI – Air quality prediction/monitoring – Machine learning – Artificial intelligence – IoT

1 Introduction

Urbanization is progressing very rapidly in many countries resulting in the production of solid waste on the increase in urban areas. This produces enormous amounts of pollutants into the air dangerously affecting human health and the environment. Africa will produce 516 million tons of solid waste in 2050 [1]. Unfortunately, in the context of Africa, the management of this waste in urban areas is usually dumped in open outdoor landfills, waste dumps for incineration of mixed and unsorted materials (domestic, medical, etc.) [2]. Incineration aims at transforming waste into ash that occupies less space. However, this can emit dangerous toxic gases [3] into the atmosphere such as $CO2$, $SO2$, $NO2$, and CO dissolved in the air, or pollute lands as ash that are worth monitoring.

The Internet of Things (*IoT*) has been proven to be very efficient in data collection for air quality monitoring systems (Indoor Air Quality and Outdoor) for urban areas. Small IoT devices embed sensors that can capture information about the environment such as pollutants. The Internet connection available in towns enables communication and data transfer to a remote Cloud. With computation resources available in the cloud, it is possible to make complex data processing as machine learning that is not possible on devices. However, IoT deployment for waste incineration in public outdoor landfills must consider many specific constraints.

The location of landfill areas is that there is very little coverage of mobile networks and the presence of white spot areas because of a low density of population in the villages and peri-urban areas [4]. These areas are generally not covered by mobile networks (*GSM*, 2*G*, 3*G*, 4*G*), or the *SMS* cost and Internet data connection for a large data collection are too high. So, the architecture for connecting the gateway to the Internet needs to take into consideration a set of constraints. The reliability and durability of electronic devices/sensors know that in addition to air pollution, there is a lot of ash that is produced that accelerates the deterioration of electronic devices. This constraint is to be put aside from the difficulties of maintenance, especially for the recalibration of sensors when environmental conditions change and since it is very dangerous to be near toxic gas fumes.

Besides, the heat of the sun and surrounding fire heat can affect sensors' reliability [5]. There is a need to detect faulty sensors for correction or predictions. Availability of electric energy under sector is not guaranteed in peri-urban and nearby rural areas. Another constraint is the expansion model of gases pollutants in the air depending on meteorological conditions.

Edge-AI is an emerging technology combining machine learning and artificial intelligence, fog and edge computing plus IoT [6]. Edge-AI has great potential to bring traditional IoT in the field of air pollution monitoring specifically from landfill incineration to the next level. Edge-AI in waste incineration pollution can provide several services based on AI such as prediction of pollutant concentrations, digital twins of sensor nodes and human organs, sensor fault detection, data correction, and federations of machine learning models for better predictions.

This chapter is a general overview of IoT-based air quality monitoring and more specifically Edge-AI technologies on urban waste incineration pollution. Section 2 describes the problem statement and impacts of air pollution from urban waste incineration, mentioning management in Africa and its effect on health and the environment. Afterward, air quality monitoring-based IoT technologies and related challenges are presented in Sect. 3. In Sect. 4, we illustrate sensor nodes deployment strategies and data transmission such as Wi-Fi, Zigbee, LoRaWAN, TV white space, etc., concerning air quality monitoring. Section 5 exhibits Edge-AI fundamentals, services, and opportunities for air quality monitoring for landfills. Finally, Sect. 6 concludes our studies and paves some future challenges.

2 Background and Impacts of Air Pollution from Urban Landfills

2.1 Urban Landfills Management in Africa

A growing waste management crisis is noticed in Africa even if the amount of generated waste is relatively low, compared to developed regions; their mismanagement impacts human and environmental health [7]. In 2012, 125 million tons per annum of municipal solid waste (MSW) were generated in Africa, of which 81 million tons (65%) were from sub-Saharan Africa [8]. By 2025, it is expected to grow to 244 million tons per year. It is worth noticing that only 55% (68 million tons) of waste is collected in Africa. Therefore, 45% of *MSW* remains within open fields, living areas, along the roads, and water points. By 2025, Africa expects to reach an average MSW collection rate of 69% [8].

The generated waste in Africa comes principally from households, commercial premises, markets, institutions, industries, and healthcare facilities [9]. Besides, a couple of transfer points are used to gather these different sources of produced waste, before urban council workers or private operators collection [9]. Following the developing countries' context, door-to-door-based waste collection is mostly used, and thus, fees are negotiated directly between customers and a given operator. Another common process collection within African countries is the “summon to bring” system. Indeed, each waste collection truck or cart has a fixed slotted time according to specific days of the week, for garbage collection. Quite often, horse or donkey carts are used for garbage collection. For instance, Fig. 1 depicts a horse-drawn carriage dumping garbage at a landfill. When the garbage carriage arrives, it hoots its horn to summon the people to bring their waste for disposal [2]. Furthermore, more than 90% of generated waste in Africa is still dumped at the side of the land, typically in uncontrolled and controlled rubbish tips [7]. Within these landfills, the methods of open disposal of household waste are burning, burying, using the waste as animal feed, or leaving the waste in the open disposal sites as illustrated in Fig. 1.

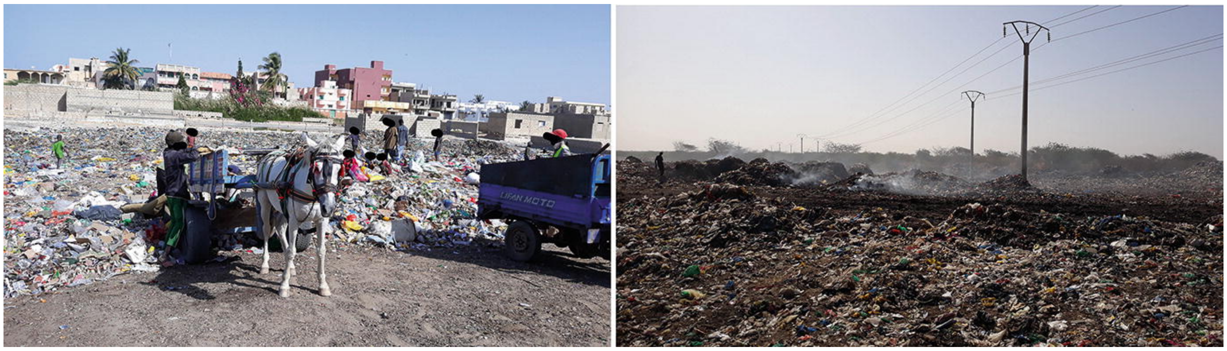


Fig. 1 A horse cart for waste collection and dumping and An open landfill incineration in the north of Senegal

This poor waste management in Africa, promoting waste open burning that generates significant air pollution, brings serious health concerns, environmental consequences, and important effects on regional and global climate change [7]. For example, open burning of waste causes significant air pollution that impacts human health and has important effects on regional and global climate change [7]. For instance, Fig. 2 illustrates a landfill located at Gandon in the Saint-Louis region, Senegal. Still, technical landfill sites are transformed into waste incineration sites without taking any prior measures for the health of populations and the environment.

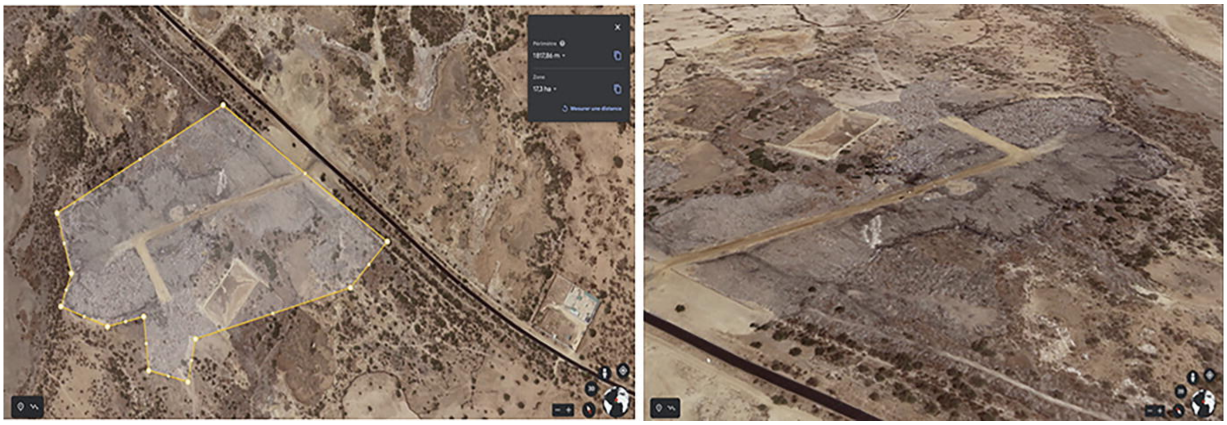


Fig. 2 Landfills incineration impact on the environment. The nearby area is full of ash and the impacted area is still expanding

Landfills in African Urban areas are generally located away from urban buildings for health reasons and to prevent the landfill from spilling over into residential areas. However, it is not uncommon to have neighborhoods or villages in peri-urban areas that are not as far from these dump areas. Landfill distances to residential zones may vary between 2 and $20\,\mathrm{km}$ for an optimized garbage collection cost. For example, in Saint-Louis region of Senegal, there is a distance of around $8\,\mathrm{km}$ between the city entrance and the landfill site, but this does not prevent the existence of illegal dumps around the city. Senegalese urban landfill areas are usually free-air deposits with or without boundary fences. Delimitation fences are generally to prevent the wind from dispersing garbage including plastic waste over a larger area.

As landfills are not treated or sorted in general, incineration without separation of the materials that are burned leads to an emanation of gas particles and ash that are harmful both to the health of the surrounding populations and to the environment nature. Chemicals and particle substances are emitted during combustion leading to air pollution. Indeed, these substances circulate in the air, dangerous when they exceed a certain threshold, and they impact human and animals' health but also the environment.

These dangerous substances are called pollutants. Table 1 depicts the main pollutants and their origin, and according to their form, they can be categorized into two groups [10]:

– **Volatile Gas Pollutants**: Carbon monoxide (CO), sulfur dioxide ($SO2$), ozone ($O3$), nitrogen oxides (NOx), volatile organic compounds (VOC), including benzene, etc.

– **Particulate Pollutants**: They are coarse ($PM10$, its diameter ranges from 2.5 to 10 µm), fine ($PM2.5$, the diameter varies from 0.1 to 2.5 µm), and finally ultrafine ($PM0.1$, with a diameter of less than 0.1 µm).

Table 1 Most common pollutants and their origins

Pollutants	**Origin**
Tropospheric Ozone (O3)	Secondary pollutant produced in the atmosphere under the effect of solar radiation by complex reactions between certain primary pollutants (NOx, CO, VOCs, etc.)
Carbon monoxide (CO)	From incomplete combustion (gas, coal, fuel oil, or wood) due to poorly regulated installations (domestic heating) or from vehicle exhaust gases
Sulfur dioxide (SO2)	Produce by the combustion of fossil fuels (fuel oil, coal, lignite, diesel, etc.) containing sulfur
Nitrogen dioxide (NO2)	Derive from the combustion of fossil fuels (heating, electricity generation, car, and boat engines)
Fine particles (PM2.5, PM10)	Come from all combustions linked to industrial for domestic activities, as well as transportation
Volatile organic compounds (benzene, butane, toluene, ethanol, acetone, etc.) VOCs	Originate from incomplete combustion of the use of solvents (paints, glues), degreasing agents, and filling automobile tank products, etc.

A former study [11] exhibits a correlation between existing diseases, like asthma and bronchitis, especially within urban regions, and poor air quality and conditions. Furthermore, the authors of [12] claim that up to 90% of Senegal's industries emissions are higher than the nation's air quality standard.

Senegal owns an Air Quality Management Center to monitor pollution in the capital. Moreover, the current measurements performed concern fixed geographic locations and are costly [13]. The dedicated equipment per site is very expensive. Nevertheless, there are more data on ambient air quality, which are acquired with low-cost sensors through air quality monitoring systems [13]. Ngom et al. [12] proposed a new air quality monitoring system that aims to consider more distributed measurements to assess air quality in real time at Dakar city. The authors have designed a hybrid low-cost measurement kit that leverages wireless sensor network technologies.

Furthermore, collected raw datasets are transmitted to a web platform for visualization and can be exploited by different applications. The measurement kit can be mobile and embedded in public transport to provide the city pollution map. The kit considered can be either fixed or embedded within a mobile car. In such cases, it is placed in the most polluted areas and can provide an idea of the evolution of air pollution in Dakar city [12]. While this is a positive prospect, the challenges associated with implementing such systems are also complex and will inevitably arise. Also, simply collecting air quality data would not be sufficient to have an efficient air pollution management system [13].

2.2 Impact of Air Pollution on Health and Environment

Pollutants cause a lot of diseases all over the world. According to [14], air pollution caused approximately 6.7 million in deaths in 2019. The diseases caused will depend on the duration of the person’s exposure to the pollutants. Thus, people with long-term exposure to pollutants are more likely to develop serious diseases. Indeed, pollutants attack different systems of the human body [10]:

- **The respiratory system**: At this level, it can be the cause of irritations, coughs, respiratory discomfort, asthma attacks, bronchitis, and pulmonary infections up to the appearance of pulmonary cancers.
- **The cardiovascular system**: Cardiovascular diseases such as myocardial infarction, angina pectoris, stroke, or heart failure can also occur due to air pollution.
- **Other Pathologies**: Air pollution can be the source of disturbances of the nervous system by the attack on cognitive functions, intoxications with strong contents involving headaches and dizziness, metabolic attacks, diabetes, obesity, or endocrine diseases.

The particles matter (PM) and nitrogen dioxide are the pollutants causing more diseases, and thus, the most dangerous for health. For PM, this is due to the fact of their small size; they can penetrate until the deepest part of the pulmonary system and even go toward other organs. In the manner of nitrogen, dangerousness comes from the fact that the air we breathe is made up of 78% nitrogen, leading to a strong combustion with oxygen ($NO2$). For this reason, nitrogen dioxide is a very present pollutant in the air. It is worth noting that $NO2$ belongs to NOx pollutants.

Apart from endangering human health, air pollution also hurts the environment, whereas acid rain causes soil and water pollution, and alteration of plants; building degradation: blackened building facades, oxidation, and corrosion of buildings; and climate change leads to the dysfunction of nature. For instance, Table 2 shows pollutant behavior across the environment. When the pollutants are produced, they can last in the environment for a long term before changing their state.

Table 2 Lifetime and pollutants propagation across the space

Pollutants	Lifetime	Spatial scale	Pollution area
O3	Several days	Regional scale	Outdoor and Indoor
CO	Few hours	Urban scale	Outdoor
SO2	Few hours	Urban scale	Outdoor
NO2	Few hours	Urban scale	Outdoor
PM2.5, PM10	Several days	Regional scale	Outdoor and indoor
VOCs	Several days	Regional scale	Outdoor and indoor

Monitoring Air Pollution is the key task for having an objective of evaluation of Outdoor Air Quality (*OAQ*) and Indoor Air Quality (*IAQ*). IoT is a commonly used technology for that task, mainly for data collection on pollutants.

---

3 IoT Technologies for Air Quality Monitoring

3.1 IoT Basic Architecture for Air Quality Monitoring

Three different layers quite often form a general IoT air quality monitoring architecture. Figure 3 illustrates the different bricks that form an IoT architecture for monitoring, as follows:

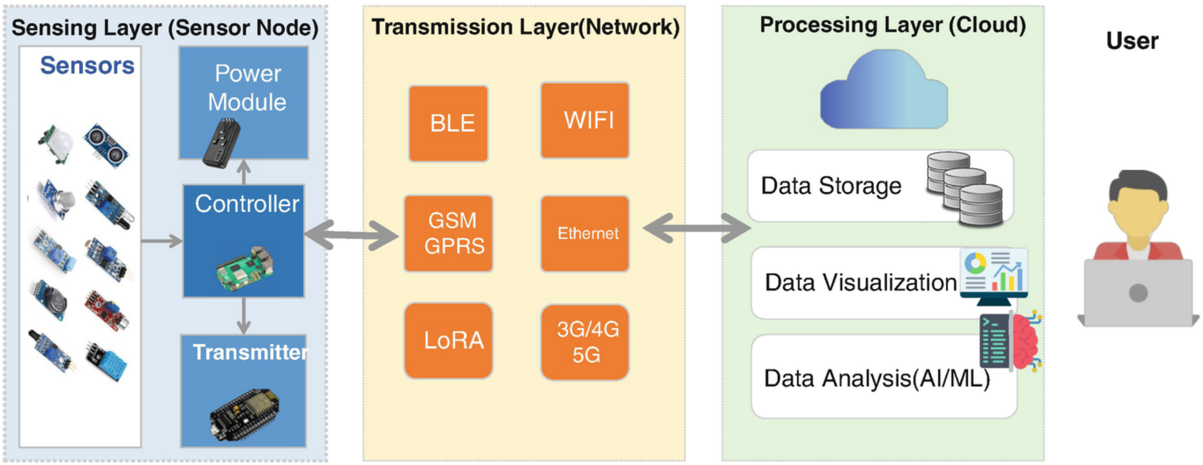

Sensing Layer (Sensor Node)
Sensors
Power Module
Controller
Transmitter
Transmission Layer(Network)
BLE
WIFI
GSM GPRS
Ethernet
LoRA
3G/4G 5G
Processing Layer (Cloud)
Data Storage
Data Visualization
Data Analysis(AI/ML)
User

Fig. 3 IoT-Based Air Quality Monitoring Architecture

- **Sensor Node Component**. This hardware part represents the sensor nodes that are responsible for collecting raw data sets. This part called the perception layer in a generic IoT architecture is also named “Sensor layer,” “Recognition Layer,” or “Device Layer” [15]. Each sensor node contains a sensor module, controller module, transmitter module, and power module [16]:

  Sensor module: Data are collected during the sensing process by sensor nodes. Sensors are selected according to the application’s requirements, and they can collect several raw datasets about location, air changes, environment, motion, vibration, etc. [17]. Air quality sensors’ efficiency and resilience are the base of all air quality monitoring systems. Raw data gathered from the sensor’s module are sent to a controller module through the interface, for example, USART [16].

  Controller module: It represents a Microcontroller Unit (MCU) that oversees receiving information from the sensor module and transmitting it toward a given destination. When the control module’s functions are implemented within an air quality monitoring system, it monitors and transmits information about air quality [16].

  Transmitter module: It ensures communication with the transmission network. Data received by the controller module are securely transmitted to gateways, if it exists, or directly to the transmission layer. Furthermore, the transmission of signals control to control devices is done through a wired or wireless communication medium [15].

  Power module: This module is represented by the batteries, solar panels or power supplies, etc. that power the sensor node. For example, a lithium battery can power a node, which can then be recharged by a solar panel. This requires the use of a solar charge controller, whose role is to lower the voltage and ensure the safety of the stored electrical energy. With the help of the solar charge controller, a battery can provide a safe and reliable power source for each module in the monitoring node [16].

- **Transmission layer**, known as the network layer, acts like a bridge between the sensor node and the processing layer. Smart objects, network devices, and networks are connected through this layer, providing functionality for network management [17]. The data collected by the sensor nodes are transmitted to the information treatment unit (or high-level decision-making units) using wired or wireless communication channels. Within the network layer, we can find an access network and a core network. Access Network gives an access environment for the perception layer with transmission technologies (GPRS, 2G/3G/4G/5G, Wi-Fi networks, ZigBee, LoraWAN, etc.) [15]. End users who are connected through the access network receive data via the core network. It establishes communication among constrained devices for resource sharing [15].

- **The processing layer**. After detection and collection, the data are sent to the processing layer through the transmission layer. The processing layer represents the last layer of the IoT architecture. At this level, intelligent computations and processing are performed; data storage and exploitation by end users are done.

  To maintain data integrity and dependability, preprocessing techniques are required to identify and change "dirty" data [16]. Thus, data are classified as valid, invalid, malicious, spamming, etc., using recognition and filtering [15]. The acquired data are then consolidated, processed, and analyzed. Analytical results are stored with the modified raw data on a storage database. When new data from the analyses become available, it is also considered. This processing enables to provide interactive services to end users [16], globally based on information collected by the sensor's nodes [15]. It provides access to personalized services and real-time decision-making applications to end users over the network according to their requirements [15]. The service depends on collected information by sensors and can therefore change for each application [17].

According to the air quality monitoring context, the next sections depict the different sensors used, selected microcontrollers for existing use cases, the means used for powering the equipment, and transmission techniques.

3.2 Sensors for Air Quality Monitoring

Landfills are mainly in two ways in sub-Saharan African cities. First, there is burial, the deposit on areas of low pressure to fill the vacuum; second, incineration that is supposed to transform these landfills into ash that occupies less space. These are the main sources of pollutants outdoor spreading. Generally, there are two types of pollutants: volatile gases and particles. Besides, one can derive two groups of sensors that are gas sensors and particle sensors:

- **Gas sensors**

  Metal Oxide Semiconductor gas sensors (MOS): These are sensors made of metal oxide surfaces that allow the concentration of specific pollutants to be known. The gas concentration is calculated by a quantifiable change in conductivity generated when the pollutant gas is in contact with the pre-heated metal oxide surface [10].
  Electrochemical sensors (*EC*): These are sensors composed of electrodes that are used to detect ambient gases through a redox reaction. This reaction, when the gas encounters the sensor, produces an electric current between the electrodes, proportional to the concentration of the ambient gas [10].
  Non-dispersive Infrared (*NDIR*) sensors: These are sensors composed of an atmospheric sampling chamber, an infrared light source, a detector, and an optical filter. When the gas is in contact with the sensor and reaches the sample chamber, then the detection is done by the emission of frequencies via an electric current (case of CO2) or by changes in the wavelength of light [5].
  Photo-ionization sensors (*PID*): These sensors are composed of electrical circuits associated with high-energy UV photons. The gas in the sensor is ionized by the UV photons, and this creates an electric current whose amount is proportional to the concentration of the gas [5].

  The *NDIR* and *PID* have low power consumption and a short response time, but they are expensive compared to MOS and EC. Indeed, according to former works on air quality monitoring, low-cost sensor solutions like MOS and EC were proposed. For instance, Table 3 exhibits several pollutants, and consequently, the *MOS* or *EC* sensors can be used for their detection.

Table 3 Pollutants and Gas Sensors

Pollutants \ Gas Sensors	**MOS**	**EC**
O3	MQ131, MiCS-2610, etc.	OX-A431, B4 series, etc.
CO	MQ-2, MQ7, MQ-9B, etc.	ME4-CO, TGS2442, B4 series, etc.
NO2	MQ135, MiCS-2714, etc.	NO2-A43F, B4 series
COV	MQ138, etc.	ZE40B-TVOC, etc.
SO2	MQ136, 2SH12, etc.	MES-SO2, 110-602 SO2, B4 series, etc.

– **Particle sensors**. According to low-cost solutions designed for air quality monitoring, different methods are used to detect fine particles. As examples, one can cite:

Light scattering method (*LSP*): The components of this method are a light source (infrared or laser), a light sensor, and an air inlet. When air enters the sensor, fine particles are captured, and light is generated from the light source. Afterward, the obtained light is scattered according to the size and particle number; it is received by a sensor photodetector that transforms the light into a signal. Therefore, the signal will estimate the number of particles in the air captured. The sensors operating on this method are of low cost and low energy consumption, but they have a limitation to the minimum size of the particles they can detect that is 0.3 μm as particles smaller do not scatter enough light [5, 10].

Scattering Size Classifiers (*DiSC*) method: This type of sensor consists of a backup filter, an air inlet, a corona charger, a scattering stage, and an induction stage. The *DiSC* detects particles by applying electrical signals on the detection surface and their presence creates physical changes. The particles go through an ionization stage, then an induction stage, and finally a diffusion stage during which an electric current is diffused according to the concentration of particles. The backup filter receives some large particles that have escaped the diffusion stage and emit a current relative to their concentration. These sensors have a high sensitivity and low energy consumption but have a high manufacturing cost [5, 10].

Digital holography method: These sensors are composed of optical signals, the air sampling channel, and the image capture system. The air sampling channel is crossed by a light generated at the source that interacts with the particles. This light is then captured by the imaging system that produces particles’ images. The number of particles is known through an image reconstruction algorithm. This method enables particle detection smaller than 0.3 µm as well as finding the particle's chemical composition. Nevertheless, the digital holography method is little used, and it provides a low sampling rate [5].

Microscopy method: These sensors are composed of the laser diode, the imaging substrate, the CMOS (complementary metal oxide semiconductor) imager, and the electrostatic particle collector. The CMOS imager generates images from an illuminated substrate. Indeed, the PM is captured by the collector who spreads them on the substrate illuminated by the laser diode. The images are then processed to have the number of particles. These sensors are used in large-volume sampling and have similar advantages to other methods. However, they have a high cost [5].

The methods are numerous and varied, but the most used is the LSP due to its low cost and meeting almost all the needs of a particle capture system. For instance, Table 4 depicts different sensors.

Table 4 Sensors and particles PM

LSP \ Pollutants	**PM0.5**	**PM1.0**	**PM2.5**	**PM4.0**	**PM10**
1528-2509-ND		●	●		●
B5W-LD0101	●		●		
SPS30	●	●	●	●	●
SM-UART-04L	●				●
PMS7003		●	●		●
480-7035-ND			●		●
OPC-N2		●	●		●

Put simply, the choice between sensors depends on deployment goals, weather conditions, and accessibility of the deployment area.

3.3 Microcontroller Market

Among electronic boards, there are two types: microcontrollers and single-board computers:

- Microcontrollers (MCU) are electronic boards with low resources compared to single-board computers. Indeed, they have low computing power and small memory capacity, a powerful chip with a processor, memory, storage, and input and output pins. The microcontroller can only execute one program at a given time, incorporates reduced system communication protocols, and has limited processing capacity [10]. For instance, one can cite Arduino Uno/Nano, Node MCU ESP32, Node MCU ESP8266, etc.

– Single-board computers (*SBC*) are small devices, with all the components of a computer, consisting of a single board with an operating system, RAM, storage, and inputs/outputs to which peripherals such as a monitor, mouse, etc., can be connected. They can play the role of a microcontroller and even more, such as the execution of several complex programs at the same time. Due to these great capabilities and the latest integrated communication protocols, single-board computers can be used as mini-servers or as a gateway for sending data to the cloud [10]. Among them, we have Raspberry Pi, Intel (Edison, Joule, Galileo), BeagleBoard, Pine A64, etc.

3.4 Data Transmission Technologies

IoT devices cannot operate without network connectivity. To enable connectivity among heterogeneous smart devices, different networking and communication technologies are used such as Bluetooth, WiFi, Zigbee, 6LowPAN, cellular networks, Sigfox, LoRaWAN, etc. [18]. Several metrics such as range, data rate, and energy consumption are used to categorize data transmission technologies [19]. According to the range, which expresses an expected maximal communication distance, transmission technologies can be categorized into 2 groups: short-range transmission technologies (Bluetooth, Wi-Fi, ZigBee, 6LoWPAN, etc.) and long-range transmission technologies (cellular networks, LPWAN, etc.):

– **Short-Range Transmission Technologies (Bluetooth, WiFi, ZigBee, 6LoWPAN, etc.).** Short-range technologies were the first attempt to provide connectivity to IoT devices. These technologies include WiFi, Bluetooth, and typical Wireless Sensor Network (WSN) approaches such as ZigBee or 6LoWPAN [20].

The main drawbacks of these former solutions are their limited coverage range, which increases either the network management complexity (e.g., multi-hop networks) or installation costs due to the need for special nodes (e.g., gateways or concentrators). Besides, the short-range standards (Wi-Fi, Bluetooth) that were developed for different IoT application purposes are not energy-aware [20].

- **Long-Range Transmission Technologies (cellular networks, *LPWAN*, etc.).** These technologies enable a transmission range over kilometers-wide distances in contrast to “short-range” standard wireless technologies. Two types of frequencies are considered: licensed (cellular network) and unlicensed ISM free band (LPWAN) [19]:

  $$\bullet$$• *Cellular technology* takes advantage of GSM/3G/4G cellular communication capabilities, and it can provide reliable connectivity, high speed to the Internet, and greater ranges covered by the base stations [21]. Thus, it is considered one of the ideal solutions for sensors-based low-bandwidth-data projects [18]. However, the solutions deployed present several issues that make them unsuitable to be directly used by constrained IoT devices [21]. An IoT cell with a huge number of devices, generating sporadic transmissions of short-length packets and gaining connectivity through a single base station, raises new challenges related to signaling, traffic control, and the reduction of consumed energy [20, 21]. Moreover, to address the challenges brought by previous cellular technologies, the next generation 5G network is available for IoT and promotes speed, coverage, and responsiveness of wireless networks but arises a new set of challenges concerning the architecture of 5G-IoT [17]. Faced with these challenges of cellular technologies, alternative solutions that provide high connectivity to end devices with low energy consumption should be deployed. Thereby, LPWAN architectures arose claiming to gather all these requirements [4].

  $$\bullet$$• *LPWAN (Low-Power Wide-Area Network)* provides very good coverage with limited data rates [22]. They promote a very low energy consumption that enables long lifetimes for IoT devices. These achievements are reached by considering the following strategies: (*i*) transmitting data

at very low bit rates; (*ii*) using low-frequency bands (sub-GHz bands); and (*iii*) limiting the communication capabilities of the end devices [20]. For instance, devices using LPWAN technologies can reach distances upper than $10\,\mathrm{km}$ with data transfer rates ranging from 0.3 to 50 kbit/s per channel. Since anyone can transmit, collisions and interference may appear within ISM bands. To enable optimal operation, the duty cycle must be configured appropriately depending on the frequency band, transmission power, and geographic location. This rigidity can be overcome by the use of free and open frequency bands with the deployment of new base stations and the possibility of peer-to-peer communication when the number of nodes is low [19]. LPWANs include Sigfox, LoraWan, and others.

The overall transmission technologies illustrated within this section can be used to monitor air pollution due to urban waste incineration. Indeed, the choice between long-range or short-range network technologies depends on the application nature and deployment environment characteristics. The actual performance of each technology regarding QoS parameters (data rate, delay, energy consumption, etc.) might be influenced by the deployment area.

4 IoT Deployment Techniques

IoT is one of the technologies used for monitoring physical and chemical parameters. In the case of air pollution monitoring, IoT devices are expanding out through target zones. Therefore, for sustainable consumer-based applications, several challenges such as low power consumption of connected material, full coverage of a targeted area, long battery life for devices, data aggregation and accessibility, low data packets collision, high network lifetime, etc., are taken into account for a successful IoT architecture deployment. An efficient node deployment aims to distribute sensor nodes in a manner that fully covers, optimizes targeted areas, and ensures node connectivity across the overall network [4].

Coverage without full connectivity will decrease the quality of IoT architecture. Indeed, with a lack of connectivity, there is no guarantee that the sink will be reached. Hence, both should be considered during the deployment [4]. An optimal deployment minimizes energy consumption while maintaining efficient coverage and connectivity. Furthermore, the presence of obstacles and weather conditions in the node's vicinity can affect the node's sensing region. Moreover, a lot of challenges are still present and issues that impinge on IoT performance.

4.1 Classical Deployment Techniques

Several techniques exist in the literature; however, we aim to describe the more classical ones that are related to our case study.

Force-Based Techniques Based on virtual forces, a given sensor node can be attractive, repulsive, or null virtual force [4]. This deployment strategy can be centralized or distributed schemes. Each sensor is subject to three types of forces. Namely, obstacles exert a repulsive force (F_{iR}) on sensors. While a favorite area coverage exerts an attractive force (F_{iA}), other sensor nodes exert attractive or repulsive force depending on the distance between them (F_{ij}) [3]. For instance, Eq. 1 expresses a net force of a given sensor node *Si*.

$$F_i = F_{iR} + F_{iA} + \sum_{j=1\&j\neq i}^{k} F_{ij} \tag{1}$$

Besides, force *Fij* between sensor *Si* and *Sj* is expressed by Eq. 2

$$\overrightarrow{F}_{ij} = \begin{cases} (W_A(d_{ij} - D_{th}), \theta_{ij}) & if\ d_{ij} > D_{th} \\ 0 & if\quad d_{ij} = 0 \\ (W_R \frac{1}{d_{ij}}, \theta_{ij} + \pi) & if\ d_{ij} < D_{th} \end{cases} \tag{2}$$

where d_{ij} denotes the distance between sensors S_i and S_j, D_{th} is a defined threshold distance value, θ_{ij} is the orientation of the line segment from sensor S_i to sensor S_j, and W_A, W_R are attractive and repulsive forces measures, respectively [23]. Several other algorithms are based on these virtual forces, including distributed virtual force algorithm (*DVFA*) [23] and Van der Waals force-based deployment [24]. The extended virtual force-based approach was proposed in [25].

Grid-Based Techniques It promotes a deterministic deployment by leveraging a geometric pattern, like a triangular grid, square grid, or hexagonal grid, where the monitoring area is split into small grids [26]. Indeed, the node's positions are fixed within these geometric areas, and sensor nodes are either in the grid center or grid vertices [4]. The grid-based sensor node deployment considers two parameters *Rs* and *Rc*. *Rc* represents the communication range of a fixed node with other sensor nodes, whereas *Rs* is the sensing range and capturing data from the environment. Thus, the ratio of *Rs* to *Rc* enables optimal deployment schemes. Some of the proposed solutions include:

– An optimal deployment [24] with ratio given by Eq. 3

$$\frac{R_C}{r_s} \geq \sqrt{3} \tag{3}$$

– An optimal sensor placement [25] with ratio depicted in Eq. 4

$$\frac{R_C}{r_s} = \sqrt{3} \tag{4}$$

– A VSGA [26] with ratio illustrated in Eq. 5

$$R_C \geq 2 + r_s \tag{5}$$

A suitable solution, within a targeted area, should verify a minimum number of nodes as enablers of an acceptable coverage and connectivity degree [4].

Computational Geometry-Based Techniques By leveraging points, line segments, and polygons, this strategy considers geometric algorithms to figure out relationships between two points or hyperplanes [4]. Even if it exists several data structures related to computational geometry, the *V oronoi* diagram and *Delaunay* triangulation act as the most used within WSNs [27]. For instance, Fig. 4 shows a deployment of both approaches [4]. Voronoi diagrams act as an integral data structure in computational geometry. By using generator points, a specific region can be regenerated into many regions [27]. Additionally, the area of interest is partitioned into several Voronoi polygons based on distances between sensor nodes. Each node is located only in one polygon, a dominant part that means a Voronoi cell [4].

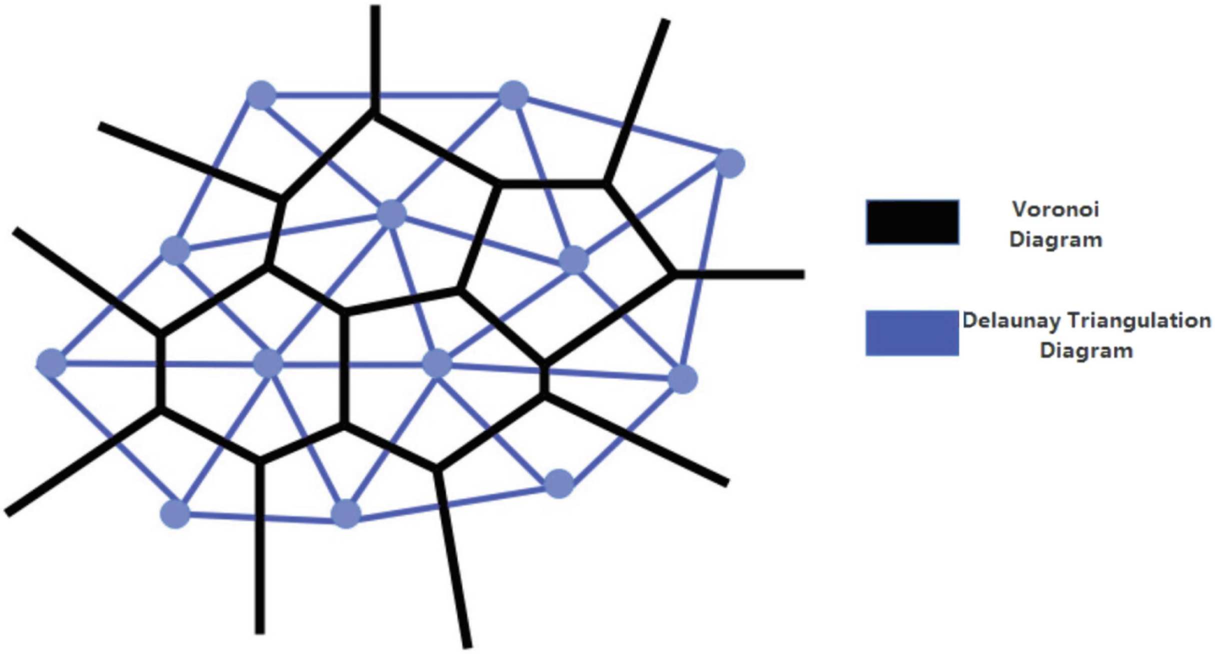


Fig. 4 Voronoi and Delaunay Triangulation Diagrams [4]

In contrast, Delaunay triangulation (*DT*) acts as the dual graph of Voronoi diagram and enables a given area to be split into various triangles that guarantee the circumcircle of each triangle contains no points [27]. Therefore, each two adjacent points within the Voronoi diagram, whose polygons share a common edge, are connected [4]. Quite often, Voronoi and Delaunay triangulation diagrams are used to eliminate or at least reduce coverage hole problems in WSNs [4]. The depicted geometric structures enable efficient IoT deployment by promoting full coverage and increasing network overall lifetime [27].

4.2 Heuristic Deployment Techniques

Self-scheduling-Based Techniques This method aims to minimize energy consumption while maintaining good coverage and connectivity. For energy saving, sleep intervals are scheduled for redundant nodes, while keeping the remaining nodes active to maintain network coverage and connectivity.

Metaheuristic-Based Techniques Since most deployment problems are NP-hard, traditional deployment algorithms give solutions that may be far away from an optimal one. Another problem is the high computational complexity of the deployment problem that is very expensive. To overcome previous drawbacks, an alternative way is to consider metaheuristic algorithms such as simulated annealing, genetic algorithm, swarm intelligence algorithm, and Artificial Bee Colony that use a meaningful guess to find an approximate correct solution [28, 29]. Unlike exhaustive algorithms that take an unreasonable time to converge, metaheuristics provide an alternative way to find efficient solutions within sustainable time.

---

5 New Trends Technologies: Edge-AI for Air Pollution Monitoring

5.1 Limitations and Challenges of Classical IoT Networks for Air Pollution Monitoring

Monitoring air quality in a landfill (an outdoor environment) involves many constraints related to the deployment environment. Following the African context, we have varying atmospheric conditions such as humid climates, dust, high temperatures, etc., which bring a big concern to IoT deployment. IoT networks operating outdoors are exposed to changing weather conditions, which may cause severe degradation in system performance [30]. For instance, Senegal, which is located within Sub-Saharan Africa, faces the same issues with sensor deployment in landfill areas. Hence, the following “important” challenges should be considered for reliable IoT-based urban air quality and waste incineration pollutants monitoring:

Network architecture and communication challenges The common position of public landfills at peri-urban areas, supposed to be sufficiently far from homes ($2{-}50\,\mathrm{Km}$ away from households) or nearby rural areas ($<50\,\mathrm{Km}$). These zones are generally not covered by a Mobile network (*GSM*, 2*G*, 3*G*, 4*G*), or the cost of SMS and Internet Data for a large data collection is too high. So, the architecture for connecting the gateway to the Internet needs to take into consideration a set of constraints. In urban areas, BLE (802.15.1), ZigBee (IEEE 802.15.4), or Wi-Fi (IEEE 802.11 b/n/g) are commonly used to connect sensor nodes to the gateway in a star topology, and Wi-Fi, wired, and Mobile networks between gateways and the cloud infrastructure.

However, in the landfills area, the IoT will need a long-range, cost-effective, and reliable link to the Internet. In the access network (sensors to gateways) Wi-Fi (IEEE 802.11 b/n/g), the best solution in terms of bandwidth has energy consumption issues. This drawback led to the use of more energy-efficient technologies such as *BLE* and ZigBee (IEEE 802.15.4), but they have low bandwidth and distance coverage. Also, network communication links reliability can be affected by different environmental factors around the landfills [30, 31]. Temperature has a major effect on the signal strength and link quality. Operations at lower temperatures might require up to 16% less power to maintain reliable communication. Rainfall also affects link transmission performance as shown in [32]. Water acts as a capacitive load due to its high dielectric constant that can modify the radiation pattern of the antenna. The signal strength at reception loses strength under these conditions. The presence of large quantities of water near the antenna has the effect of detuning it, thus reducing its effectiveness.

Sensors Reliability The reliability of sensors in IoT networks is to be monitored during Air Pollution Monitoring of landfills because of the environmental conditions. These environmental conditions affect sensor accuracy and can alter sensor measurements. Waste incineration areas are generally open areas and therefore exposed to the sun temperature almost all day. Temperatures can go up to 50°C under the bare Sun. With fire, the ambient temperature is up to $60C$ near landfills. Sensors are also exposed to continuous ash deposits over the long term, which can lead to the sensors becoming increasingly unreliable and less efficient. With all these constraints, it becomes important to be able to automatically detect when a sensor is no longer reliable or defective or to be able to correct the data of a poorly calibrated sensor. More technically, temperature, humidity, and wind direction are key factors that affect the sensor’s performance [5]. Sensors are sensitive to temperature changes, and the heat can result in significant inaccuracies in air quality measurements. A combination of low humidity and high temperature is harmful to some sensors and can damage them.

Energy and long-term data processing Real-time monitoring with IoT implementation also requires an in-depth analysis of power requirements, energy efficiency, additional hardware, and maintenance needs. Therefore, it is critical to be able to guarantee sensor operations in a continuous and long term without random erroneous data. The IoT-based *IAQ* monitoring systems are further required to send real-time updates to the end users regarding pollutant concentrations. The *MCUs* generally have limited storage capacity that is not enough for handling extended real-time monitoring. Long-term monitoring can pose an additional challenge in terms of repeated repair and replacement of faulty products. Depending upon how many pollutants are required to be monitored from the target site, the *IAQ* monitoring system may need multiple sensor nodes. Based on the number of monitoring parameters, researchers also need to be careful about *MCU* (Microcontroller Unit) selection and data storage requirements [33].

All these constraints lead to the necessity to find additional technology to compensate for the risk of reliability of collected data or decision-making information. Integration of IoT for massive data collection and AI for processing and prediction opens new opportunities for scientific topics. Edge-AI is one of these approaches that combine IoT, Fog-Edge Computing, and AI.

5.2 Edge-AI Fundamentals

Edge-AI has emerged and has gained significant attention with the latest innovations in artificial intelligence (AI) and Machine Learning (more precisely), new computation architecture (Fog–Edge) efficiency, and the proliferation of IoT devices. Edge-AI refers to the practice of doing AI computations near the users at the network's edge instead of a centralized location like a cloud service provider's data center or a company's own private data center. With Edge-AI, machine learning models are deployed into a smart device or near the sensors [34]. Edge-AI can also help establish a relatively optimal routing strategy, perform congestion control by learning the dynamic network state, and provide fast responses to users. The deployment of edge servers with large computing and storage capabilities can greatly improve the performance of these technologies in IoT [35].

Edge-AI is a paradigm that can support significantly the development of real-time applications, thanks to decentralized processing close to the data source systems (such as local servers and IoT systems). It can also reduce the amount of data transmitted to the cloud by selecting relevant data by filtering them by segmenting images, data fusion, …The necessary bandwidth and data storage requirements will be reduced, while the security and privacy, and confidentiality of data will be improved (private data will remain on private area).

Edge-AI also relies on the data and computation distribution model (with decentralization), to offer resilience to failures, reduction of latency, and reduction of bandwidth costs. Distribution means that end systems (IoT or Gateways) will do part of AI tasks so that the system can work at a certain level of service without using functionalities from the cloud.

The IoT applications area on industries on which Edge-AI can have a big impact are diverse: air quality monitoring, smart homes and cities, autonomous cars, digital health, etc. [34]. Edge-AI, in these contexts, will deploy equipment that can make local smart decisions based on local AI models that can make complex predictions based on their computation capacities.

First-stage AI occurs locally at the edge device rather than sent to the cloud to improve processing efficiency in this localized data [36]. Furthermore, it enables building reliable systems where service intelligence is provided at the node level, allowing other computer devices to act as backups and ameliorating performance degradation by reducing service downtimes [37]. For instance, Fig. 5 depicts a basic Edge-AI architecture where data acquisition, data transmission, and data analytics bricks are illustrated.

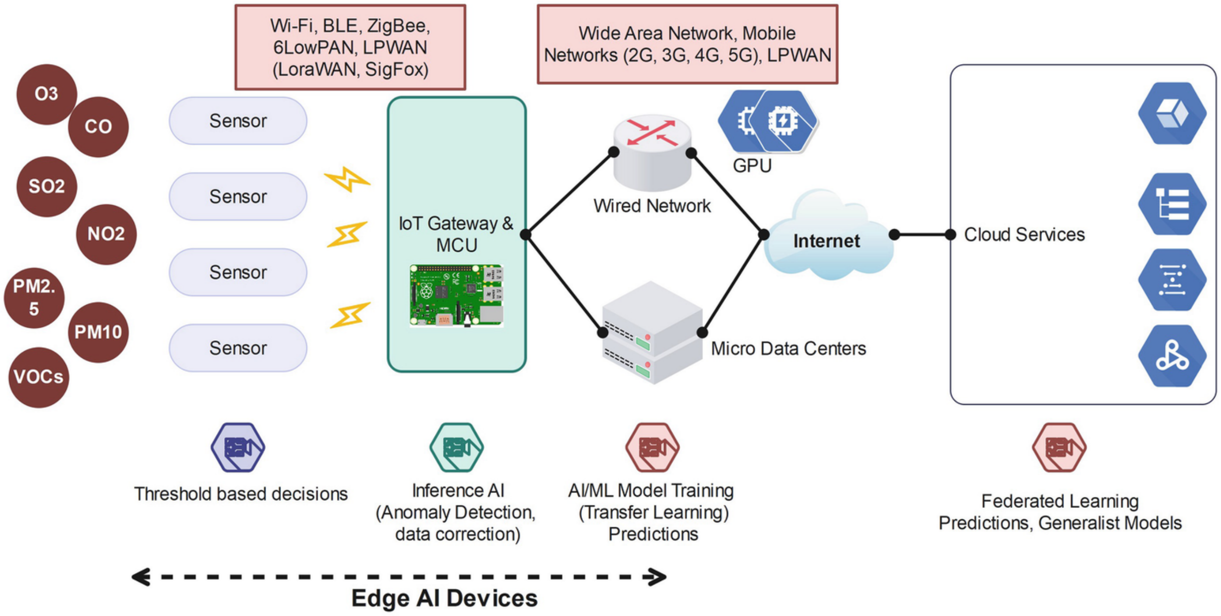

Wi-Fi, BLE, ZigBee, 6LowPAN, LPWAN (LoraWAN, SigFox)
Wide Area Network, Mobile Networks (2G, 3G, 4G, 5G), LPWAN
O3
CO
SO2
NO2
PM2.5
PM10
VOCs
Sensor
Sensor
Sensor
Sensor
IoT Gateway & MCU
GPU
Wired Network
Internet
Micro Data Centers
Cloud Services
Threshold based decisions
Inference AI (Anomaly Detection, data correction)
AI/ML Model Training (Transfer Learning) Predictions
Federated Learning Predictions, Generalist Models
Edge AI Devices

Fig. 5 Basic Edge-AI architecture

Basically, Edge-AI includes hardware and software that enables AI algorithms to localize and operate on edge devices [34]. The Edge-AI hardware part is composed either of a mini data center of less than 10 servers or of single servers placed especially to execute a reduced number of models and artificial intelligence algorithms which we call cloudlets, or of gateways reinforced by *GPUs* that can run artificial intelligence or machine learning micro-models. Key AI technologies development Edge-AI relies on are the following:

- **Edge devices**: These Edge-AI nodes must have hardware processing power (based on CPUs and GPUs), sensors, and storage capacity. These hardware nodes can be micro data centers (local servers), IoT nodes, and/or gateways with GPU.
- **Machine learning frameworks**: ML frameworks, such as TensorFlow Lite and PyTorch, are used to develop and run machine learning models on edge devices.
- **Deep Learning and ML**: Artificial neural networks (CNN, RNN, GAN, etc.) have now established themselves in the field of machine learning and AI. In addition, they are flexible enough today to adapt to different types of problems including those that can be found in EDGE Computing.
- **Model compression**: The compression of models also allows its reduction to be embedded in equipment such as smartphones. In the context of Edge, the localized nature of the data and the specialization of the task intended for the AI may be relevant to remove from the model all data that do not apply to the specific context of the Edge-AI application.
- **Federated learning**: Federated machine learning enables different devices to participate in distributed machine learning without having to share local data. The results can be obtained by an aggregation of models resulting from the learning of local data at each local machine.
- **Edge-to-cloud integration**: Integration of the Edge and the cloud enables local processing at the Edge level for tasks that do not require large resources and the use of cloud services for more complex tasks such as training on large data sets or large model deployment.

All these advances in technologies enable new applications and use cases while improving the performance, efficiency, and security of Edge-AI.

5.3 Machine Learning Framework for Air Quality/Pollution Prediction

Machine learning methods are used in different ways in several research projects in the field of air quality.

Some of the most famous methods are:

- **Linear and Multi-linear Regression**: Regression algorithms build a relationship between two or more explanatory variables (independent) and a response (dependent) variable by fitting a linear equation to the data. Multiple linear regression (*MLR*) requires some assumptions, such as normality of error distribution, homogeneity of variance, and independence of errors [38]. *MLR* is a statistical method allowing the prediction of a variable based on two or more parameters. This is a linear regression in $\mathbb{R}^n$ with $n > 1$. The predicted variable is called the dependent variable, and the parameters are called independent or explanatory variables.

  $$\hat{y} = \epsilon + \sum_{i=1}^{n} (\beta_i \times x_i) \tag{6}$$

  where β_i are regression coefficients, x_i are values of explanatory variables values, and ϵ is the model's random error (residual) term.

- **Support Vector Machine**: *SVM* algorithm maps data into a high-dimensional feature space via a non-linear relationship and then performs a linear regression within this space [38]. *SVM* tries to fit the error within a certain threshold. The goal of *SVM* is to discover the role $f(x)$ with a condition that $f(x)$ is within a required accuracy ϵ from the value *yn* of every data point. *SVM* is also known as tube regression.

  $$f(x) = \sum_{n=1}^{N} (a_n - a_n^*)\left(x_n^T x\right) + \mathrm{b} \tag{7}$$

  The input quality vector is *x*, the space limit is *b*, as well as a_n are familiarized large range multipliers.

- **Random Forest** *RF* is a regression technique that combines the performance of numerous *DT* algorithms to classify or predict the value of a variable [38]. This method can be used for classification or regression purposes. *RF* is a powerful prediction method that can be applied in multiple research fields, such as social sciences or environmental sciences, among others [38].

$$H(X) = \frac{1}{T} \sum_{i=1}^{T} h_i(x) \tag{8}$$

where *T* is the number of regression trees and $h_i(x)$ is the output of the $i-th$ regression tree (h_i). Therefore, the output of the Random Forest is the average of the output values of the decision trees.

- **Artificial Neural Networks**: *ANN* has become the mainly used Machine Learning model in non-linear problems [38]. *ANN* models are a computational technique inspired by the biological neural system and can be used for different purposes such as prediction, clustering, or pattern recognition. *ANNs* are formed by artificial units called neurons. *ANN* is formed by input, intermediate, and output layers [38].

  Deep learning algorithms such as convolutional neural networks (*CNNs*) and recurrent neural networks (*RNNs*) can learn automatically features from the raw inputs, removing requirement for manual feature engineering. Also, a Recurrent Neural Network (*RNN*) is a type of ANN that deals with sequential data or time series data. Examples of famous *RNN* are bidirectional recurrent neural networks—*BRNN*, Long Short-Term Memory—*LSTM*, and Gated Recurrent Units—*GRUs*.

$$\mathrm{p} = \mathrm{f}\left(\sum_{\mathrm{i}=1}^{\mathrm{m}} \mathrm{w}_{\mathrm{i}} \mathrm{x}_{\mathrm{i}}\right) + \mathrm{b} \tag{9}$$

AI/ML is used to perform different tasks related to landfill air pollution monitoring and management such as air pollutant prediction, Forecasting Air Pollution Particulate Matter, Air Quality Index (AQI), the impact of air pollution on health, and identification of emerging pollutants [39, 40].

Machine learning is also used to predict the atmospheric concentration of pollutants over time, individually or in groups of pollutants. This prediction aims to prevent risks to human health linked to these pollutants. The main pollutants for which concentrations are commonly predicted in the air pollution topic are carbon monoxide (*CO*), ozone (O_3), sulfur dioxide (SO_2), carbon dioxide nitrogen (NO_2), nitrogen oxides (*NOX*), and particles ($PM2.5,\ PM_{10}$) [41]. The relationship or impact of pollutants on health and the quantification of deaths caused by their high concentration are also estimated using the tools offered by machine learning

[41]. The fourth use concerns the characterization of pollution sources according to the pollution sources. The used dataset provides mainly three categories of data as predictor variables: pollutant variables (NO_2, *CO*, O_3, $PM_{2.5}$, PM_{10}, SO_2), weather variables (*TEMP*, *PRES*, *DEWP*, *RAIN*,*WD*, *WSPM*), and spatio-temporal variables (year, season, month, week, weekday, hour, geolocation) [41].

LSTMs/RNNs are also used in air quality prediction models due to their effective performance in time series. Several papers explored using LSTM-based models to predict AQ pollutants. *ARIMA* algorithm combined with numerical modeling to forecast $PM_{2.5}$, *O*3, and *NO*2. Hybrid methods (LSTM-CNN, LSTM-Linear Regression, KNN-MLP, ARIMA-Random Forest) are also used in this context. Multiple linear regression (MLR) machine learning model has been used to predict the concentration of $PM_{2.5}$ and NO_x. In [42] LSTM and CNN have been used in dioxin emissions prediction and have interesting results for dioxin control. The raw datasets consist of 618 records (samples) of online dioxin emissions data from this incinerator [41]. Ding et al. [43] show that GBDT (gradient boosting decision tree) gives a more precise prediction of NO_x emission of waste incineration.

In monitoring air quality, SVM, RF, ANN, RNN, and Fuzzy methods can be efficient in predicting when a fixed threshold is reached [38]. Machine learning models are sometimes compared (*DT*, *RF*, *SV M*, *Bagging*, *MLP* or *LSTM*, *SV R*, $Naive-Bayes$, and *KNN*) to see the best one for different tasks. The most commonly seen as the best algorithms for *AQI* predictions are *LSTM*, *SV M*, $RandomForest$, and Hybrids [41] (Fig. 6).

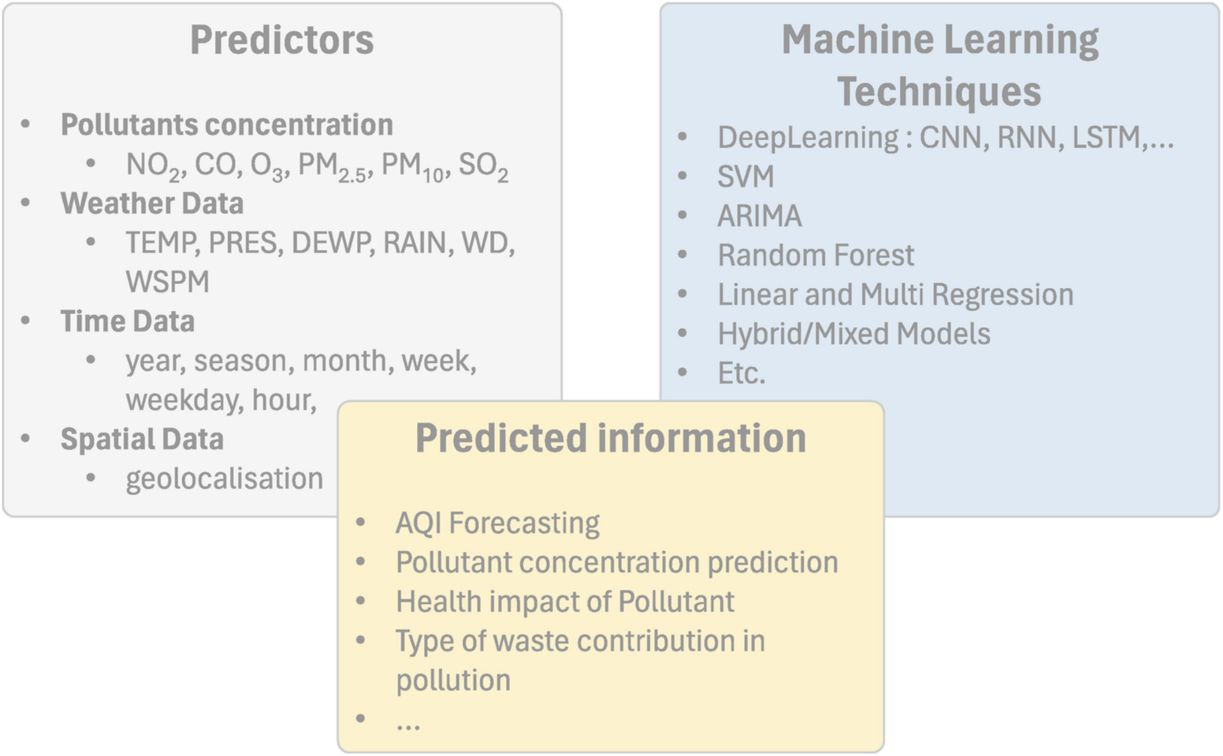

Predictors
Pollutants concentration
NO2, CO, O3, PM2.5, PM10, SO2
Weather Data
TEMP, PRES, DEWP, RAIN, WD, WSPM
Time Data
year, season, month, week, weekday, hour,
Spatial Data
geolocalisation
Machine Learning Techniques
DeepLearning : CNN, RNN, LSTM,...
SVM
ARIMA
Random Forest
Linear and Multi Regression
Hybrid/Mixed Models
Etc.
Predicted information
AQI Forecasting
Pollutant concentration prediction
Health impact of Pollutant
Type of waste contribution in pollution
...

Fig. 6 Machine Learning Framework for Air Quality/Pollution Prediction

5.4 Edge-AI Opportunities in Urban Waste Incineration Pollution Monitoring

IoT facilitates Air Pollution Monitoring with environmental data collection (gases). Edge-AI extends these functionalities and brings new AI-based opportunities. On one hand, Edge-AI enables local site/intermediate nodes' decision-making, prediction, or inference capabilities that open new applications such as pollution early warning systems. On the other hand, Edge-AI, by bringing more or less complex AI tasks to devices/nodes close to data sources, is a way to mitigate the lack of connectivity toward the cloud, and network latency due to low bandwidth in waste disposal areas. Deploying AI models on the Edge nodes of Urban waste Incineration pollution Monitoring Systems instead of in the cloud allows Edge-AI devices to analyze data faster, local faster decision-making, data security improvement, and user experience quality enhancement (interactivity). Furthermore, Edge-AI can help designing and deploying smart systems that can perform tasks such as decision-making, problem solving, pattern recognition, and learning adapt to changing environments, and learn from experience without being explicitly programmed. For example, in the urban waste incineration pollution monitoring systems context, it can enable to make preventive decisions and predict dangerous air pollution concentrations. By combining IoT and AI/ML functionalities, the Edge-AI has several avenues for innovation and solutions to the limits of traditional IoT in the field of air pollution monitoring specifically from landfill incineration:

- **Autonomous Intelligent Monitoring**: Edge-AI can enable autonomous operations, as it can be used in hostile environments such as waste incineration sites. Autonomous operations include decision-making in the event of danger by issuing alerts. This reduces the need for human personnel while improving efficiency. IoT sensors with cutting-Edge-AI can monitor toxic gas emissions, detect dangerous situations, and alert people who need to take action. They can also collect data and predict the concentration of dangerous pollutants to anticipate potential danger. Edge-AI can also allow monitoring stations with humans to be replaced with virtual monitoring stations [44].
- **Design of Early Warning Systems** [9]: Early warning systems are defined as *UNISDR* to be “The set of capacities needed to generate and disseminate timely and meaningful warning information to enable individuals, communities, and organizations threatened by a hazard to prepare and to act appropriately and in sufficient time to reduce the possibility of harm or loss”. Concerning monitoring air quality, EWS may involve detecting the first emanations of toxic particles from incineration zones or illegal dumping of dangerous chemicals in an open area.
- **Landfills pollution impact prediction**: One of the opportunities for Edge applications and at the level of air quality monitoring is the prediction of the impact of pollution in fact in incineration zones there can be several gas emissions more or less dangerous. However, before raising alerts, a system is needed that is capable of predicting the extent of the area that will be impacted by pollution. If the pollution is confined to an area that is not inhabited and which is unlikely to have any impact on populations or environments, the alert level will not be the same if there is wind, which allows transport of toxic agents to the surrounding peri-urban homes. In a classic IoT system, the deployment of sensors without intelligence would not allow this prediction to be made. The data will be sent to the cloud level which contains the machine learning tools.

With the Edge, it becomes possible to have data that are collected on the area of the Edge processing resources allowing predictions to be made on a well-localized area since the data will not be very large. Thus the risk assessment can be done live and more frequently.

- **Waste Characterization**: Another opportunity that Edge-AI offers for air pollution management is the characterization of waste. Indeed, in African urban and peri-urban areas, sorting garbage is very complicated, as is evaluating the quantity of waste produced by populations. While this information is very important depending on the types of waste that are collected, treatments could be differentiated and thus avoid the incineration of products that can cause the release of toxic gases. Artificial intelligence and machine learning techniques can be used to characterize the types of materials that are in landfills as well as their quantity from the type of gases that are produced with the incinerator. This would make it possible to assess the level of risky pollution at any time depending on the periods and the collection carried out.

Edge-AI enables AI/ML models on edge devices such as smartphones, IoT devices, and robots, without relying on cloud computing.

5.5 Edge-AI Challenges

The deployment of Edge-AI systems in Urban Air Pollution from Waste incineration is not without challenges. Moreover, several challenges must be overcome to successfully implement Edge-AI systems implementation. Here are some potential open technical and operational issues for the future of Edge-AI:

Technological Challenges Some of the traditional challenges of IoT need to be enhanced, especially [34]: Computing Resources (running large and growing AI models in fewer resources devices than cloud); Internetworking (transmitting large amounts of data and machine learning models to and from Edge-AI nodes with limited network resources, connectivity, and bandwidth); Deployment architectures and protocols; real-time Data processing at the edge (data fusion, validation, and completion); Energy efficiency (model compression to reduce the energy consumption of AI algorithms); Data privacy (when related to health) or Edge-to-cloud integration. For example, the cloud can be used for training large and complex machine learning models, while the edge nodes are used for real-time inference. It is essential to enable system and network managers of cloud resources to easily create optimized configurations for machine learning applications. The needs of Edge-AI applications have different needs in terms of storage and computing resources. However, the cloud service for Edge-AI must be flexible enough to provide resources as needed at the time of deployment and operation of the system [38].

Urban Waste Incineration pollution monitoring requires an efficient deployment of reliable sensors in the area of interest. This brings many challenges to efficient real-time data processing and analysis. Edge-AI deployments provide many of the necessary mechanisms, e.g., by offering faster data rates, energy-efficient networking, and having support for computing and data storage closer to the sensors. These improvements open new directions to design cutting Edge-Air quality monitoring systems [5].

Deploying Edge-AI systems effectively by considering aspects such as computer resources, internetworking, data confidentiality and security, real-time operational needs, and selecting the appropriate tools and technologies unlocks new possibilities for realtime decision-making as well as enhanced confidentiality and safety [34].

Operational challenges of AI deployment The implementation of the Edge for monitoring air quality pollution also presents many operational challenges of AI deployment, but these challenges can be grouped into different categories:

Deployment of AI models The first option is to deploy them on the sensor nodes; however, this solution involves finding ways to extend the computing capabilities of the sensor nodes and therefore finding sufficient energy sources to be able to deploy many nodes. The second track is to have an Edge node, a server, or a mini buffer data center not far from the place 2 collections and which is connected to the gateway. This solution is the most realistic because it makes it possible to have the data close to their places of collection at the same time but also to have enough energy and computing power to be able to store and execute them. However, it sometimes requires the deployment of infrastructure when the landfill is quite far away.

The $3rd$ solution is to have the data at the cloud level. It is the classic solution, but it does not bring any added value compared to areas that have Internet connection difficulties. The acquisition price and the cost of energy mean that this solution can only be achieved by an operator who has an activity that revolves around that.

Specialized and localized vs. general and federated AI models Indeed, the data concerning landfills in an urban area can have quite different realities from other areas depending on lifestyles and sources of pollution. For example, urban areas with industries producing a lot of waste with various manufactured products-based pollutants will not have the same reality or the same AI model as a semi-urban region and therefore whose waste has more biological pollutants. The federation of machine learning models allows at the same time to federate models to have a larger and more generalist model but heavier to be executed at the edge and therefore requiring more resources. On the other hand, these federated models should be deployed with specific parameters for each area simply. Therefore, transfer learning can be used to have pre-initialized models that will be gradually specialized over the data collection and that will also allow updating the general federated model.

Dimensioning of the computing resources that are affected according to the monitoring over the long term or even over the medium term. Indeed, if the monitoring is done over a very long period, the machine learning model will grow and at some point cannot be run on an Edge server but rather on data centers that will decentralize.

Prediction of the extent of pollution with the concentrations knowing that it is very complicated to have a sufficiently dense sampling of data of impacted areas by landfills incineration pollution. Machine learning algorithms like deep learning need to have the most detailed data possible to be able to function accurately. While for this collection, in most cases the monitoring can be very localized, which leads to an extrapolation of the surrounding areas. Finally, the use of these data can be used either to raise alert levels or to have alarms when critical thresholds are exceeded.

---

6 Conclusion

In Africa and more deeply in Senegal, waste incineration causes a lot of pollution issues. Therefore, it is mandatory to implement incineration areas effective monitoring systems that protect the surrounding populations. The urban waste incineration pollution assessment helps with decision-making regarding potential risks to the environment as well as living beings’ health. For a sustainable environment, promoting health, and attaining well-being by reducing waste incineration, air quality monitoring is vital for the planet.

Therefore, we highlighted the impacts of urban waste incineration pollution from urban landfills since 90% of generated waste in Africa is still dumped at the side of the land, typically in uncontrolled and controlled rubbish tips.

We depicted urban waste management in Africa, with a focus on Senegal, as well as the main emitted pollutants coming from waste incineration and their impact on the population’s health and the environment. Furthermore, our study exhibits the diseases caused by pollutants and the lifetime and pollutants propagation across space. As a use case, we plan to figure out the impact of waste incineration at Gandon landfill located in the Saint-Louis region, Senegal as well as the nearest boundaries or areas vicinity.

Therefore, air quality monitoring with low-cost sensors is an alternative to the costly professional air quality monitoring stations. Indeed, IoT architecture is the most used technology where sensors are expanding out through targeted areas that are either indoor/outdoor or white spot areas. It is worth noting that for an efficient IoT deployment, several challenges such as a harsh environment, low power consumption of connected material, full or partial coverage of the targeted area, long battery life for devices, data aggregation, and delivery should be overcome. For this reason, we reviewed the different IoT deployment and transmission technologies by taking several metrics.

By leveraging the latest innovation in cloud, fog, and edge computing to conduct data analytics as close as possible to the data collected point, IoT systems aim to mitigate a couple of issues such as latency, coverage, lack of connectivity, unconnected areas, etc. Consequently, this survey depicts new technologies such as Cloud/Fog/Edge Computing and Edge-AI, which are the trending technologies in the IoT world. The main objective of Edge and Fog Computing is to move computing resources out of centralized data centers to be closer to devices. The capabilities of such concepts could be extended by introducing various machine learning algorithms, for instance, Edge-AI, paving the way for sustainable and secure futuristic designs. Edge-AI implementation brings a mix of benefits and challenges that are also highlighted in this chapter.

Nevertheless, cybersecurity concerns and challenges have not been described within this survey even if they represent open research issues from existing systems and need to be considered by researchers in future systems.

References

1. Kaza, S., Yao, L.C., Bhada-Tata, P., Van Woerden, F.: What a Waste 2.0: A Global Snapshot of Solid Waste Management to 2050. World Bank, Washington, DC (2018). https://doi.org/10.1596/978-1-4648-1329-0
2. Adebayo Bello, I., bin Ismail, M.N.: Solid waste management in Africa: a review. Int. J. Waste Resour. **6**(2) (2016). https://doi.org/10.4172/2252-5211.1000216
3. Li, C., et al.: Identification of emerging organic pollutants from solid waste incinerations by FT-ICR-MS and GC/Q-TOF-MS and their potential toxicities. J. Hazard. Mater. **428**, 128220 (2022). https://doi.org/10.1016/j.jhazmat.2022.128220 [Crossref]
4. Farsi, M., Elhosseini, M.A., Badawy, M., Arafat Ali, H., Zain Eldin, H.: Deployment techniques in wireless sensor networks, coverage and connectivity: a survey. IEEE Access **7**, 28940–28954 (2019). https://doi.org/10.1109/ACCESS.2019.2902072 [Crossref]
5. Concas, F., et al.: Low-cost outdoor air quality monitoring and sensor calibration: a survey and critical analysis. ACM Trans. Sens. Netw. **17**(2), 20:1–20:44 (2021). https://doi.org/10.1145/3446005
6. Asha, P., et al.: IoT enabled environmental toxicology for air pollution monitoring using AI techniques. Environ. Res. **205**, 112574 (2022). https://doi.org/10.1016/j.envres.2021.112574 [Crossref]
7. Godfrey, L., et al.: Solid waste management in Africa: governance failure or development opportunity? In: Edomah, N. (ed.) Regional Development in Africa. IntechOpen (2020). https://doi.org/10.5772/intechopen.86974
8. Scarlat, N., Motola, V., Dallemand, J.F., Monforti-Ferrario, F., Mofor, L.: Evaluation of energy potential of Municipal Solid Waste from African urban areas. Renew. Sustain. Energy Rev. **50**, 1269–1286 (2015). https://doi.org/10.1016/j.rser.2015.05.067 [Crossref]
9. Okot-Okumu, J.: Solid waste management in African cities “East Africa”. In: Marmolejo Rebellon, L.F. (ed.) Waste Management—An Integrated Vision. InTech (2012). https://doi.org/10.5772/50241
10. Mokrani, H., Lounas, R., Bennai, M.T., Salhi, D.E., Djerbi, R.: Air quality monitoring using IoT: a survey. In: 2019 IEEE International Conference on Smart Internet of Things (SmartIoT), Tianjin, pp. 127–134. IEEE (2019). https://doi.org/10.1109/SmartIoT.2019.00028
11. Toure, N.O., et al.: Observed and modeled seasonal air quality and respiratory health in Senegal during 2015 and 2016. GeoHealth **3**(12), 423–442 (2019). https://doi.org/10.1029/

2019GH000214
[Crossref]

12. Ngom, B., Seye, M.R., Diallo, M., Gueye, B., Drame, M.S.: A hybrid measurement kit for real-time air quality monitoring across Senegal cities. In: 2018 1st International Conference on Smart Cities and Communities (SCCIC), Ouagadougou, pp. 1–6. IEEE (2018). https://doi.org/10.1109/SCCIC.2018.8584551
13. Mir Alvarez, C., Hourcade, R., Lefebvre, B., Pilot, E.: A scoping review on air quality monitoring, policy and health in west African cities. Int. J. Environ. Res. Public. Health **17**(23), 9151 (2020). https://doi.org/10.3390/ijerph17239151
[Crossref]
14. Fuller, R., et al.: Pollution and health: a progress update. Lancet Planet. Health **6**(6), e535–547 (2022). https://doi.org/10.1016/S2542-5196(22)00090-0
[Crossref]
15. Gupta, B.B., Quamara, M.: An overview of Internet of Things (IoT): architectural aspects, challenges, and protocols. Concurr. Comput. Pract. Exp. **32**(21) (2020). https://doi.org/10.1002/cpe.4946
16. Paithankar, D.N., Pabale, A.R., Kolhe, R.V., William, P., Yawalkar, P.M.: Framework for implementing air quality monitoring system using LPWA-based IoT technique. Meas. Sens. **26**, 100709 (2023). https://doi.org/10.1016/j.measen.2023.100709
[Crossref]
17. Burhan, M., Rehman, R., Khan, B., Kim, B.-S.: IoT elements, layered architectures and security issues: a comprehensive survey. Sensors **18**(9), 2796 (2018). https://doi.org/10.3390/s18092796
[Crossref]
18. Yaqoob, I., et al.: Internet of Things architecture: recent advances, taxonomy, requirements, and open challenges. IEEE Wirel. Commun. **24**(3), 10–16 (2017). https://doi.org/10.1109/MWC.2017.1600421
[Crossref]
19. Foubert, B., Mitton, N.: Long-range wireless radio technologies: a survey. Future Internet **12**(1), 13 (2020). https://doi.org/10.3390/fi12010013
[Crossref]
20. Sanchez-Gomez, J., Sanchez-Iborra, R., Skarmeta, A.: Transmission technologies comparison for IoT communications in smart-cities. In: GLOBECOM 2017—2017 IEEE Global Communications Conference, Singapore, pp. 1–6. IEEE (2017). https://doi.org/10.1109/GLOCOM.2017.8254530
21. Al-Sarawi, S., Anbar, M., Alieyan, K., Alzubaidi, M.: Internet of Things (IoT) communication protocols: review. In: 2017 8th International Conference on Information Technology (ICIT), Amman, pp. 685–690. IEEE (2017). https://doi.org/10.1109/ICITECH.2017.8079928
22. Gerodimos, A., Maglaras, L., Ferrag, M.A., Ayres, N., Kantzavelou, I.: IoT: communication protocols and security threats. Internet Things Cyber-Phys. Syst. **3**, 1–13 (2023). https://doi.org/10.1016/j.iotcps.2022.12.003
[Crossref]

23.
Priyadarshi, R., Gupta, B., Anurag, A.: Deployment techniques in wireless sensor networks: a survey, classification, challenges, and future research issues. J. Supercomput. **76**(9), 7333–7373 (2020). https://doi.org/10.1007/s11227-020-03166-5
[Crossref]

24.
Yu, X., Liu, N., Huang, W., Qian, X., Zhang, T.: A node deployment algorithm based on Van Der Waals force in wireless sensor networks. Int. J. Distrib. Sens. Netw. **2013** (2013). https://doi.org/10.1155/2013/505710

25.
Li, J., Zhang, B., Cui, L., Chai, S.: An extended virtual force-based approach to distributed self-deployment in mobile sensor networks. Int. J. Distrib. Sens. Netw. **2012** (2012). https://doi.org/10.1155/2012/417307

26.
Al-Turjman, F., Hassanein, H., Ibnkahla, M.: Quantifying connectivity of grid-based Wireless Sensor Networks under practical errors, p. 223 (2010). https://doi.org/10.1109/LCN.2010.5735706

27.
Devi, S., et al.: The use of computational geometry techniques to resolve the issues of coverage and connectivity in wireless sensor networks. Sensors **22**(18), Art. no 18 (2022). https://doi.org/10.3390/s22187009

28.
Tsai, C.-W., Tsai, P.-W., Pan, J.-S., Chao, H.-C.: Metaheuristics for the deployment problem of WSN: a review. Microprocess. Microsyst. **39**(8), 1305–1317 (2015). https://doi.org/10.1016/j.micpro.2015.07.003
[Crossref]

29.
Verma, A., Ranga, V., Angra, S.: Relay Node Placement Techniques in Wireless Sensor Networks (2020). https://doi.org/10.36227/techrxiv.11454228.v1

30.
Luomala, J., Hakala, I.: Effects of temperature and humidity on radio signal strength in outdoor wireless sensor networks. présenté à 2015 Federated Conference on Computer Science and Information Systems, pp. 1247–1255 (2015). https://doi.org/10.15439/2015F241

31.
Boano, C.A., Tsiftes, N., Voigt, T., Brown, J., Roedig, U.: The impact of temperature on outdoor industrial sensornet applications. IEEE Trans. Ind. Inform. **6**(3), 451–459 (2010). https://doi.org/10.1109/TII.2009.2035111
[Crossref]

32.
Markham, A., Trigoni, N., Ellwood, S.: Effect of rainfall on link quality in an outdoor forest deployment. In: International Conference on Wireless Information Networks and Systems, pp. 148–153 (2023). https://doi.org/10.5220/0002959201480153

33.
Saini, J., Dutta, M., Marques, G.: Internet of Things for Indoor Air Quality Monitoring. SpringerBriefs in Applied Sciences and Technology (2021). https://doi.org/10.1007/978-3-030-82216-3

34.
Singh, R., Gill, S.S.: Edge-AI: a survey. Internet Things Cyber-Phys. Syst. **3**, 71–92 (2023). https://doi.org/10.1016/j.iotcps.2023.02.004
[Crossref]

35. Lv, J., Shi, Y., Cheng, H., Lin, Z., Ma, L.: Guest editorial: AI and edge computing driven technologies and applications. Digit. Commun. Netw. **9**(2), 448–449 (2023). https://doi.org/10.1016/j.dcan.2023.03.003
[Crossref]

36. Whaiduzzaman, M., et al.: A review of emerging technologies for IoT-based smart cities. Sensors **22**(23), Art. no 23 (2022). https://doi.org/10.3390/s22239271

37. Tuli, S., et al.: AI augmented Edge and Fog computing: trends and challenges. J. Netw. Comput. Appl. **216**, 103648 (2023). https://doi.org/10.1016/j.jnca.2023.103648
[Crossref]

38. Iftikhar, S., et al.: AI-based fog and edge computing: a systematic review, taxonomy and future directions. Internet Things **21**, 100674 (2023). https://doi.org/10.1016/j.iot.2022.100674
[Crossref]

39. Doreswamy, H., Harishkumar, K.S., Yogesh, K.M., Gad, I.: Forecasting air pollution particulate matter (PM2.5) using machine learning regression models. In: Procedia Computer Science, Third International Conference on Computing and Network Communications (CoCoNet'19), vol. 171, pp. 2057–2066 (2020). https://doi.org/10.1016/j.procs.2020.04.221

40. Livingston, S.J., Deepa Kanmani, S., Shamila Ebenezer, A., Sam, D., Joshi, A.: An ensembled method for air quality monitoring and control using machine learning. Meas.: Sens. **30**, 100914 (2023). https://doi.org/10.1016/j.measen.2023.100914

41. Méndez, M., Merayo, M.G. Núñez, M.: Machine learning algorithms to forecast air quality: a survey. Artif. Intell. Rev. **56**, 10031–10066 (2023). https://doi.org/10.1007/s10462-023-10424-4
[Crossref]

42. Wen, C., et al.: Dioxin emission prediction from a full-scale municipal solid waste incinerator: deep learning model in time-series input. Waste Manag. J. **170**, 93–102 (2023). https://doi.org/10.1016/j.wasman.2023.08.004
[Crossref]

43. Ding, X., Feng, C., Yu, P., Li, K., Chen, X.: Gradient boosting decision tree in the prediction of NOx emission of waste incineration. Energy **264**, 126174 (2023). https://doi.org/10.1016/j.energy.2022.126174
[Crossref]

44. Samad, A., Garuda, S., Vogt, U., Yang, B.: Air pollution prediction using machine learning techniques—an approach to replace existing monitoring stations with virtual monitoring stations. Atmos. Environ. **310**, 119987 (2023). https://doi.org/10.1016/j.atmosenv.2023.119987
[Crossref]

OceanofPDF.com

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
S. Pal et al. (eds.), *IoT Edge Intelligence*, Internet of Things: Technology, Communications and Computing
https://doi.org/10.1007/978-3-031-58388-9_13

Enabling Artificial Intelligence on IoT Edge: Smart Approaches and Solutions for Providing Remote Dental and Medical Services

Aya Sedky Adly[1], Afnan Sedky Adly[2] ✉, Eve Malthiery[2, 3, 4],
Shahid Ali Shah[2], Elias Estephan[2, 5] and Mahmoud Sedky Adly[2]
(1) IRMB, University Montpellier, INSERM, Montpellier, France
(2) LBN, University Montpellier, Montpellier, France
(3) CSERD, CHU Montpellier, Montpellier, France
(4) UFR odontologie, University Montpellier, Montpellier, France
(5) Neuroscience Research Center, Faculty of Medical Sciences, Lebanese University, Beirut, Lebanon

Keywords Artificial intelligence – Internet of Things – Remote dental services – Remote medical services – Edge computing – Telemedicine

1 Introduction

One of the major issues in many developing nations of the world, in which a large number of people live in rural areas, is establishing fairness in healthcare access for all people worldwide. There are numerous obstacles to providing fair medical treatment in these areas. In their surrounding areas, there is a reported shortage of healthcare services, including doctors, dentists, and diagnostic tools [1]. The majority of the established government primary care centers are significantly understaffed as a result of many physicians’ reluctance to practice in distant areas [2].

The majority of patients in these areas must travel for most of the day to get to any of the nearby hospitals, which is a huge barrier to getting frequent screenings and exams. Taking vacations to go to the hospital costs money for the rural population, who are mostly relying on their daily incomes. Even when patients do arrive at a hospital, it is sometimes the case that health care givers are already very busy to schedule any time for consultation because of the high patient load and excess numbers [3].

The newest addition to the computing paradigm is edge computing. Edge computing's primary goal is to reduce transmission delays through overall power consumption and network backhaul by getting cloud-centric services closer to the end users. IoT is a strong option to work with edge computing to enhance the existing ubiquitous solution's intelligence [4].

According to current research, the healthcare industry is the most difficult because it requires sophisticated exploitation of a wide range of computing and application development platform capabilities [5].

A group of tools utilized in healthcare applications and services that can connect using the most recent communications technologies is called Internet of Medical Things (IoMT). It can help reduce hospital visits and the enormous strain on a medical system by enabling health data transmission across a secure communication network and connecting individuals to their healthcare providers. The IoMT industries have smart electronics devices that could be used on a body, at home, in society, or in hospitals and are specifically made for the health care sector. Those devices could offer real-time data for location, health issues, and other crucial details and services [6, 7].

Over the past several years, as a result of developing computer-based sophisticated technology, emerging disease prevention strategies, and enhanced diagnostic techniques, dentistry has undergone a full transformation. Internet of Dental Things (IoDT) is considered a cutting-edge strategy to manage periodontal diseases, dental caries, oral cancer, and other dental diseases. IoDT may be crucial in monitoring and collecting information about a patient for oral healthcare, and this information may also be used for future risk research and assessment [8].

Clinical dentistry has undergone a significant transition over the past decades due to the developments in digital technology, the rapid evolution of dental materials, enhanced clinical patient-treatment strategies, and the introduction of patient-centered preventive-treatment measures [9]. Clinical dentistry today has greatly benefited from the digital revolution, and this trend will undoubtedly continue in the years to come [10].

In the meantime, the healthcare industry begins to use edge computing to enhance healthcare services. For various types of healthcare analysis, the healthcare platform installed in smart homes can enable the usage of mobile devices and sensors, besides increasing data storage and processing capability [11, 12].

It allows for the efficient use of resources, faster access to the storage and processing infrastructures, and the sharing of analytical results [13].

AI-based health management software, smart devices, and other components are all designed to fulfill patient needs and provide whatever medically relevant information is possible, enabling patients to get medical care in the most efficient and suitable way [14]. All partners in this networked healthcare system are “connected” as a result of the prompt sharing of reliable patient data. It strives to ensure that patients, physicians, and the family members always have access to crucial data, making it a crucial part of the future of healthcare. It can be seen that IoMT has offered a comprehensive vision in this new era by utilizing smart devices and processing enormous quantities of complex information to offer the best and most numerous services. Conversely, because IoT devices are so interconnected and frequently provide connected health services, it can also be said that each edge device has a storage burden and is computationally heavy [15]. This problem can be solved by taking into account how AI integration and powerful edge computation can provide intelligent edge network acquisition of enormous complex data and computation services for essential professional analysis [16, 17].

Edge computing reaps a number of advantages by enabling processing close to the data source and avoiding needless data transportation, such as quick service delivery and efficient network operation to guarantee the quality of services and enhance a user experience [11]. In order to shorten reaction times and save money on bandwidth, more services are currently being moved from a cloud back to a network’s edge. In the overall computing paradigm, the function of edge computing devices has changed from data consumers to data producers/consumers due to the widespread use of IoT and mobile devices having more powerful compute capabilities [18].

The edges in IoMT have been found to be capable of continuously delivering sophisticated services to clinicians, carers, and health care providers from any location at any time. In addition, it has been determined that the Edge AI’s solution is superior to the cloud AI’s [19]. In comparison to cloud AI, it is less expensive in terms of data communication costs, privacy risks, and power consumption because it does not require a connection. Therefore, creating wearable, versatile devices that are better at simplifying healthcare could be beneficial. In addition, healthcare can function intelligently by strategically responding to complicated system data updates [20].

2 Edge Computing

One of the major goals of edge computing is locating resources adjacent to consumers and sources of information in order to combat the known drawbacks of cloud computing, including excessive jitter, latency, and limited bandwidth.

Both in the context of the IoT and computing devices of the user, these performance metrics are crucial and pertinent for wireless access networks [21].

2.1 Reduced Delay

In applications with high processing, such as gaming and augmented reality, where mobile consumers anticipate uninterrupted service from the content providers, it is crucial to reduce the amount of time needed for a packet to go from a node to the destination [22].

2.2 Increasing Services Agility

Fixed IoT sensors and nomadic mobile devices produce massive amounts of information that are sent to a central cloud. However, because of the Cloud’s centralized design, several crucial components, such as contextual and geographical awareness, have been missing. If processing of edge computing data is at the edge networks, then location awareness and context would be much more achievable and attainable [23, 24].

2.3 Improving Resources Management

In edge computing, the management of resources plays an important role in the assignment of tasks, which are produced by the local device to a remote cloud and are the center of networks or local devices/servers. A significant approach to resources management in edge computing is by assigning the tasks to local servers or remote cloud,taking into consideration a number of factors like bandwidth consumption, energy, and latency [25]. Those approaches could be assigned based on their goals to categories including task completion time reduction, energy consumption reduction, and system throughput improvement [26, 27].

For example, a study proposed a prediction model to dynamically estimate resource use depending on users' behavior and allocate resources accordingly. An application offloading decision-making technique based on the application's features was proposed by other studies [28]. In order to adapt to changes in network bandwidth and latency, a study presented a resource management technique to assess the mobile platform's energy usage and offload computation tasks. Moreover, in order to reduce the time needed for an application to complete, another study introduced the online task management algorithms, which take into account the general task dependency on a mobile device [29–31].

2.4 Removal of Single Point of Failure

Edge computing contains a limited quantity of resources, which allow applications to manage processing, networking resources, and offloading in order to attain high levels of performance and efficiency. In addition, edge computing architecture allows a distributed technique, "instantaneously" rerouting traffic to backup edge network resources in the event that the primary pool of edge network resources fails [32, 33]. This is made feasible by cutting-edge technology like network function virtualization (NFV) and software-defined networking (SDN), which also improve network interaction with current IoT settings [34].

2.5 Consistency

Either on the edge network or the local device, architecture of Edge Computing could decide where to offload data. Smart sensors make decisions, which enhances network performance overall and ensures that only important data is sent to the cloud. For instance, a closed-circuit television (CCTV) camera only records and sends information when something moves right in front of it [35, 36].

2.6 Distinctive Qualities of Edge Computing

Cloud computing technology is present in several edge computing aspects. However, the following distinctive qualities make edge computing a potential technology.

- Utilizing Mobility

With the rise in mobile device usage, mobility support is now more important than ever. Edge computing supports mobility by adhering to the location ID separation protocol (LISP). The LISP protocol uses a distributed directory system to uniquely identify the host and the location. The primary method for enabling mobility in edge computing is the separation of location identity from host identity.

- Utilizing Geographical Distribution

Edge computing installs multiple information processing platforms at the network's edge, bringing the benefits of the cloud closer to the users. The following are some benefits of these geographic allocations:

(a) The edge systems, such as sensor networks, provide extensive real-time analytics assistance.

(b) Network administrator can offer location-based mobile service without traversing via the entire WAN.

(c) Big data processing and analysis could be done more accurately.

- Utilizing Proximity

Popular consumer applications like video streaming services and user-generated video demand a lot of bandwidth or low latency, like augmented reality and video chat. These services are responsive due to edge computing's close proximity.

The proximity of data processing and analysis services to edge devices enhances user interaction and response. The proximity of these services to the user [14] allows for improved off-loading decisions by providing context information. Similar to this, service providers can assess and enhance their offerings based on customer input regarding resource allocation [37, 38].

- Utilizing Navigation and Location Awareness

Another crucial feature that enables edge computing to improve location-based services is the location-awareness attribute. For many medical computer applications, real-time location data is necessary [39].

2.7 Medical Automation and Sensors

Wearable, ingestible, and embedded sensors are used by medical automation systems to support user medical and dental care. They encourage wellness by continuously monitoring the user's health indicators and giving timely treatment or rehabilitation with the appropriate dosage. These systems could be categorized into four groups according to their goals: remote rehabilitation systems, remote consultation systems, remote diagnosis systems, drug infusion systems, and remote monitoring systems [1].

- Remote Rehabilitation Systems

After a major incident, such as a stroke, traumatic injury, arthritic conditions, etc., rehabilitation systems are used to enhance the user's physical capabilities. These tools support the user, give honest feedback, and lessen the need for hospitalization. The wearable rehabilitation robotic hands are an example of rehabilitation systems. According to a study, a robotic hand can help with repeated hand motions, offer feedback, and support a user's efforts. Pneumatic artificial muscles and a proportional-integral-derivative controller are used to achieve their design. Through comprehensive experiments, the efficacy of these rehabilitation systems has been confirmed [40].

- Remote Consultation Systems

The scope for remote consultations was greatly expanded by the advancement of computer technology in the 1980s. Today, digital image processing is a major component of programs for remote consultations. Equipment for medical examinations can either provide video signals that can be converted to digital images or immediately output digitized images.

Workstations in radiology are available for conventional X-ray exams, computer tomography, and magnetic resonance.

The remote consultation subject might be direct treatment of a patient, teaching, or instruction, where the health personnel are existent at both locations [41].

There is a lot of medical imaging equipment available, but not all of it generates images that are suitable for direct tele-transmission. For instance, analogue film is produced by X-ray equipment. Applications used for remote consultations are therefore somewhat varied. In some specialties, still images are sufficient, but live images are sometimes needed. While some examinations can be filmed and broadcast as video clips, others necessitate the specialist's teaching and direction. There are several levels of resolution, color or grayscale needs, and picture dynamics. Additionally, digital and analog visual sources are available. The selection of tele-technical solutions is constrained by all these diverse demands [42].

- Remote Diagnosis Systems

Remote diagnosis is a medical practice where a health care provider concludes a diagnosis based on transmitted data from the patient. Remote consultation is considered an expansion of this concept by the fact that it also involves the treatment of patients, and furthermore, it is not limited to health care providers [1].

- Drug Infusion Systems

Drug infusion systems are able to deliver exact doses of medication into the body through implanted and electronically controlled infusion pumps. These devices can be utilized to give insulin to diabetic patients, transmit morphine for the treatment of chronic pain, and deliver antineoplastic therapy to patients with liver cancer [43, 44].

- Monitoring Systems

Systems that allow for remote, real-time observation of patients' symptoms and other health-related outcomes could provide cost-effective solutions for enhancing remote medical and dental care. In order to support clinical decision-making, related systems have been applied to the treatment of numerous chronic illnesses. The need for more research has been emphasized, particularly to clarify which patient populations and conditions warrant remote monitoring and wearable sensors in order to provide patients and healthcare professionals with the greatest value possible [45].

2.8 Enabling of Edge AI

In contrast to using central cloud servers for processing, edge AI denotes the deployment of artificial intelligence approaches and algorithms for devices that are at or closer to a network edge, such as mobile devices, sensors, or Internet of Things (IoT) devices. This can make it possible to analyze data more quickly and effectively while also enhancing security and privacy. Examples of real-time information processing approaches at the edge for systems in which processing speed is essential like streaming video and music, and online communication [46].

Additionally, Edge AI can be given the computational intelligence necessary to create an intelligent system capable of performing activities like learning, pattern recognition, problem-solving, and decision-making that ordinarily need human-level intelligence. Machine learning, swarm intelligence, evolutionary computation, fuzzy systems, and neural networks are just a few of the subfields that make up computational intelligence [47]. Applications including optimization issues, control systems, data mining, and robotics frequently make use of these techniques. Computational intelligence aims to develop intelligent systems that can learn from experience and adapt to changing contexts without explicit programming. When direct communications between machines or devices are necessary without the need for human interventions, the machine-to-machine (M2M) approach may be taken into account [48]. Numerous wired and wireless communications technologies, including Bluetooth, cellular networks, and Wi-Fi, can be used to implement this strategy. Remote monitoring and control systems and smart home automation are a few examples of M2M applications [49].

---

3 Enabling Technologies for Medical and Dental Healthcare Services Using Artificial Intelligence

3.1 Edge Computing

Edge computing, as it is commonly known, enables computing directly at a network’s edge, enabling the delivery of novel services and applications to a huge number of connected devices. Access points, set-top boxes, cellular base stations, units installed along roads, etc. are typical examples of edge devices. Edge computing, on the other hand, could be thought of as an extension of cloud computing, that takes place on the networks edge, where numerous decentralized, ubiquitous, homogeneous, and heterogeneous devices interface with one another to perform storage and information processing. Mobility at a network's edge, which is the foundation for service delivery in the medical and dental healthcare systems, is a core property of edge computing [50].

3.2 Semantic and Cognitive Computing

IoT applications that are clever and intelligent must come up with strategies for using machine-interpreted information for the decision-making process. They need to modify themselves to different conditions and ideas using machine learning or other artificial intelligence approaches. A greater understanding requires various ontologies for various topics. Semantic technologies have the potential to raise the level of reasoning used by IoT components and improve system and application compatibility [51].

3.3 Semantic Representation of Data and Semantic Edge Computing

One of the key areas of artificial intelligence is semantic representation and analysis. In the area of IoT, uniform semantic web representations can be used. Semantic edge refers to a deep comprehension of machine-to-machine communications in addition to machine-to-human communications. It utilizes logical principles to decode reasonable information for the appropriate provision of healthcare services in the area of medical and dental information exchange [52].

At the level of edge computing itself, this is possible. The platform doesn't need large amounts of data on a regular basis for healthcare services. A system must be event-based, and data collection must be query-based [53].

3.4 Knowledge Management (KM) and Ontologies

Information is organized using ontologies powered by artificial intelligence. They can also be applied to formally represent knowledge.

In order to facilitate access to and coordination of knowledge assets, ad hoc frameworks called Knowledge Management Systems (KMS) have been created throughout the past two decades. KMSs are data management systems that boost knowledge operators' productivity by managing organizational knowledge [54].

Ontologies make it possible for represented knowledge to be shared, combined, and reused. W3C Semantic Web standard web ontology language (OWL) is considered one of the knowledge representation languages that can share and provide knowledge in a form machines could understand utilizing their own parsers. The Open Geospatial Consortium (OGC), Domain Working Group, Sensor Web Enablement (SWE), and the Semantic Sensor Networks (SSN) Incubator Group have made special efforts to standardize and provide high level ontologies for suitable services integrations in order to facilitate sensor network interoperability [55]. A well-known study on the application of semantic technology in IoT demonstrated the significance of semantic interpretation for IoT system interoperability [56].

3.5 Privacy and Security

It is essential to use a secure information computation method in which outsourced data are communicated in ciphertext form in order to create safe healthcare systems and avoid privacy and security issues with the edge paradigm.

Given the urgency with which privacy and security preservation have become issues for edge computing in the process of replacing cloud-computing, significant advancements in artificial intelligence (AI) have been made, which open numerous avenues for addressing the encountered security concerns. The AI systems can recognize harmful assaults more precisely and effectively because of AI's strong learning capabilities. And at the same time, the transfer of model parameters rather than raw data also helps to prevent privacy leaks up to a certain extent [57].

3.6 Body Area Network (BAN)

In order to facilitate a range of applications, such as entertainment, personal, medical, consumer electronics, and others, the IEEE 802.15 working groups were presented with standard communications protocols that are suitable for devices of small power and suitable operation in, around, or on human bodies. Bio-medical sensors, wireless audio transmissions, sports equipment, and personal electronics all employ these BAN technologies. Each of the aforementioned applications has certain demands regarding bandwidth, power consumption, latency, and distance. The device's primary function is to provide ubiquitous nature and allow for connectivity with the existing IP-based networking [58].

4 Internet of Medical Things and Artificial Intelligence

The Internet of Medical Things (IoMT) indicates medical devices that have an ability to communicate information over networks without the need for human to computer or human-to-human interactions.

4.1 IoMT and Smart Systems

An IoMT platform is a smart system that is primarily made up of electronic circuits and sensors to collect biomedical signals from patients, processing units to process the signals, network devices to transmit the biomedical data over networks, permanent or temporary storage units, and visualization platforms with artificial intelligence schemes to make decisions at the physician's convenience [59].

Due to its capacity for processing massive quantities of data, controlling processes, and producing precise results to produce the best possible outcomes, artificial intelligence (AI) has attracted researchers and biomedical sectors attention. Artificial intelligence (AI) can perform cognitive activities such as sophisticated imaging, knowledge-based learning, guided surgery, and logical reasoning. In this approach, AI is helping the healthcare system by reducing effort, avoiding unnecessary hospitalizations and inaccurate diagnoses, and saving patients' time and money by avoiding pointless appointments. Developing cutting-edge healthcare solutions requires bridging the gap between AI-assisted systems and nanotechnology interfaced with IoMT technologies [59].

Hospitals that employ intelligent, optimized, and automated modules (which may be grounded in AI/ML) on the information and communication technology infrastructure to improve patient care procedures and offer innovative features are known as "smart hospitals." Telemedicine, telehealth, and remote robot surgery are just a few of the many applications for smart hospitals. While telehealth focuses on providing remote, non-clinical care, telemedicine focuses on delivering clinical treatments at a distance. In remote robot surgery, medical robots do operations under the supervision of a surgeon who can be located at far distances [60].

4.2 Emerging Technologies in IoMTs

Physical Unclonable Function (PUF) Devices

Massive security and confidentiality risks are posed by the IoMT systems' accessibility and unpredictability.

As a portable and reliable way to secure IoMT devices, Physical Unclonable Functions (PUFs) are suggested. The key benefit of a PUF over current traditional cryptography methods is that it has the ability to work with IoMT devices that have limited computational sources [61].

Mobile Edge Node (MEN)

Between the cloud and data sources, a MEN implements intermediary processing and storage tasks. The MEN combines non-medical and medical data from various sources, processes the obtained data in-network, performs categorization and emergency notifications, extracts data of interest, and then sends the extracted or processed data to the cloud. Essentially, the MEN can be used to construct a variety of healthcare-related applications, including long-term chronic disorder management. These applications can assist patients in participating actively in their care and in conveniently interacting with their physicians whenever and wherever they choose. A MEN that runs specialized context-aware processing also makes it simple to link and maintain numerous data sources close to patients while improving data transmission depending on a context. (i.e., patient's state, data types, and supported applications) and wireless network conditions [62].

Blockchain Technology

New approaches to healthcare data management, sharing, access control, retrieval, and storage are emerging because of blockchain technologies. Researchers are concentrating on employing blockchain tactics to add security to healthcare applications as a result of the development of blockchain technologies.

Since the IoMT business is growing quickly and is a data-intensive industry, a lot of sensitive data needs to be kept safe from manipulation. Blockchain is a digital ledger that is peer-to-peer and has undergone security testing. Direct contact between people who don’t trust each other is made possible by blockchain. Security architectures based on IoMT that utilize blockchain to safeguard data transmission between connected nodes are considered challenging [63].

---

5 Internet of Dental Things (IoDT) and Artificial Intelligence

5.1 Current Dental Issues

A vital component of overall health is oral health. According to the most recent World Health Organization (WHO) reports, oral disorders are one of the most common illnesses in society and can result in discomfort, pain, tooth loss, and, in rare instances, even death. With an estimated four billion victims worldwide, dental caries is the most common dental illness. Additionally, dental illnesses rank as the 11th most prevalent illness worldwide. Oral cancer is considered one of the most common malignancies in the population and is also considered the most common cancer overall. Therefore, oral cancer, periodontal disease, and tooth caries are prevalent conditions in today’s society [45]. Dental procedures are time-consuming and expensive, accounting for 5% of global health care spending. In underdeveloped nations, oral health expenses account for 20% of total spending. Modern diets high in fermentable carbohydrates, cigarette and alcohol use, poor oral hygiene practices, and a lack of fluoride exposure all contribute to dental problems, including dental caries and periodontal diseases [64].

5.2 Importance of IoDT

The Internet of Medical Things is an advanced cloud network technology that has been effectively applied in the healthcare industry. The IoT technology has been used mostly in medicine to prevent new diseases, manage chronic diseases, and actively monitor patients’ conditions.

The Internet of Dental Things (IoDT) will significantly improve oral disorders such as periodontal disease, dental caries, and oral cancer detection and prevention. IoDT will be crucial in the collection and follow-up of data in the oral healthcare system, and it can support dentists in developing novel techniques for risk assessments.

Dental caries is considered a complex illness in the tooth’s biofilms that results in the breakdown of organic and inorganic materials in both hard and soft tooth structures as a result of a mineral imbalance. The primary caries-causing factors in the caries process are lowered pH, regular ingestion of sugary diets, or decreased salivary secretion in the oral cavity, caries-causing bacterial flora (such as lactobacilli and streptococci mutants), time, and impaired oral hygiene condition in a patient. Periodontal diseases are inflammatory conditions that affect the gingiva, the bones that surround the teeth, and the periodontal ligaments. Therefore, periodontal disorders and dental caries are chronic bacterial illnesses that might lead to tooth loss [65].

As a result of lifestyle, behavioral, and dietary changes, systemic disorders side effects, and patients’ weakened oral health approach, dental caries and periodontal disorders remain common even nowadays.

Lesions in the oral cavity are being invaded by oral malignancies, which are the sixth most prevalent kind of cancer in people. Early oral cancer lesions detection and diagnosis are essential for effective cancer therapy. Most patients with oral cancer today have late-stage detection, which results in a poor prognosis and disastrous surgery outcomes [66].

Early diagnosis and prevention of the oral malignancies are very important tasks for dentists. Because of the rapid paradigm shift in management modalities, expanded medical researches, improved living standards, and an increase in patient longevity over the past several years, future therapy modalities will focus on disease prevention. “Prevention measures are as important as Cure methods” will serve as the foundation for the futuristic approach to dental treatment. Future innovations in technology, such as IoDT, will aid in the effective control and prevention of oral disorders such as tooth caries [67].

5.3 Future Management

The future therapy for dental caries as well as other oral illnesses can be centered on three things: risk assessment, early disease detection, and prevention. Those preventative interventions need to be simple to implement for a broad population, effective for usage at the community level, and affordable for patients [68].

Therefore, IoDT has a lot of potential to spread in the various areas of dentistry because of how simple it is to use. IoDT offers a technological link between the patient and the practitioner of oral health care, enabling quick analysis, transfer, and collection of patient data.

In conclusion, internet of things in the dentistry would have a bright future and could expand dental research’s boundaries.

6 Challenges and Solutions

6.1 Heterogeneous Information

In order to monitor patients effectively, healthcare organizations use a variety of information sources. Therefore, a promising trend toward remote monitoring and automating supervision tasks is the use of modern multimodal data processing algorithms to combine multiple information sources at the edge. However, there are still a number of problems with healthcare systems that use hybrid sensing sources. First, in healthcare systems where power consumption is a limiting concern, it is not easy to include and transmit many data streams across multiple modalities. For battery-powered devices, the transmission of highly informative biosignals is in fact an energy-hungry process. Second, signal artifacts can be caused by external sources such as noise, signal offset, and interference, as well as internal sources like muscle movements and activities, which have a major impact on the data quality.

Adopting a healthcare system design in this situation would be advantageous since it enables the system complexity brought on by such diverse and variable data-stream inputs to be addressed [69]. This is accomplished by utilizing multimodal in-network processing techniques, which produce both the cross-modal correlations and the temporal correlations inside each modality. Additionally, an architecture enables the extraction of high-level application-based features on the edge as opposed to a cloud. Therefore, instead of delivering the compressed data or original data, a MEN could send a small number of the obtained correlations or the extracted features. Second, to enhance signal quality before to transmission, advanced signal processing for artifact removal could be incorporated on an edge [62].

6.2 Allocation

In IoMTs, the number of packets is enormous compared to normal offloading conditions, and the data size of the monitored healthcare packets is very little. Additionally, IoMTs need a lot of concurrency. Priorities should be assigned to various users according to their needs. Because of high time complexity, conventional spectrum allocations techniques are therefore inadequate to handle such complex and interdependent scenarios. Intelligent spectrum allocation algorithms must distribute wireless channel resources in an adaptable manner without using deterministic rules. The energy consumption and latency can be greatly decreased without central management [70].

6.3 Scalability

Another significant problem that develops as more devices are connected to a single cloud is scalability. When multiple devices attempt to access cloud services at once, a congestion could result. Additionally, uploading all data from all end devices could occasionally result in inefficient bandwidth use, particularly if segments of bandwidth-intensive data like video streams would be uploaded but are useless in deep learning activities [71].

6.4 Cyberattacks

Several cyberattacks, involving model poisoning attacks, inferential attacks, and concerns of privacy can cause the framework to be vulnerable.

Moreover, Edge devices themselves can be susceptible to several physical attacks or denial-of-service (DoS) attacks. Therefore, securing the model from those attacks is a crucial study field [72].

6.5 Utilizing and Implementing Dynamics for Healthcare

The majority of currently published works concentrate on semi-static or static healthcare scenarios in which the arrival and distribution of healthcare information are predetermined. It is believed that decision-makers are fully aware of the precise details of medical analytic tasks (e.g., the required CPU cycles to complete tasks, and data size) and compute capabilities of edge servers [73]. However, in practice, edge servers and medical analytic tasks are considered heterogeneous, making it challenging to precisely acquire them. Additionally, the task execution time is not linearly dependent on the resources used for calculation, which makes it more difficult to estimate from historical data. Natural language processing, computer vision, and other fields have all greatly advanced through machine learning. However, it is still difficult to figure out how to use it for modeling stochastic processes and implementing dynamics in healthcare [74, 75].

6.6 Bandwidth Limitations

With the accentuated growth in the number of connected edge devices, their generation of information is also accentuating tenfolds. Connected devices may produce tens of megabytes of information, including the speed and route of the linked device, its working circumstances, background data (movement and weather), camera frames, and safety movies. Even more data are anticipated to be produced by self-operated devices. High network bandwidth is needed to transport such enormous amounts of information to the cloud. However, it is occasionally limited for a variety of reasons, including transmission regulations and privacy issues [76].

6.7 Data Bias

While less regulated or restricted data are being added through local sources, there is a significant chance that bias will be introduced. Although there are various methods for reducing bias in machine learning models, more studies are still required. Similar to this, it is important to investigate the effects of corrupted or incorrectly labeled data, as well as the data provenance [77].

6.8 Managing of Information and Energy

Numerous studies have looked into energy harvesting as a potential solution to the IoMTs’ unstable energy source issue. It is anticipated that as communication technologies advance, energy will be given alongside tasks including medical analysis [78]. To re-model the transmission rate and data sizes in such circumstances, new transmission protocols are necessary. Additionally, edge servers’ energy dissipation can be greatly decreased, but the associated energy queue also has to be remodeled. It is also important to take into account how various IoMTs interact with one another. Because users are not distributed similarly, different IoMTs have different burdens. In order to realize load balancing, overloaded edge server can send too many tasks to nearby idle server.

Similarly, the longevity of sensors in the body in overloaded IoMTs can be increased via wireless power transfers, which can create power balance among IoMTs. Joint transmissions of information and energy could lower equipment maintenance costs while also increasing spectrum efficiency [74].

6.9 Devices Limited Capabilities

Since the majority of Edge devices have limited resources, it is possible that these devices could quickly run out of computational capacity and storage. As a result, fresh approaches to optimizing and managing the usage of computation and local storage in this setting may be inspected. Data collectors, sensors, controllers, medical implants, and security cameras are just a few examples of IoT devices with restricted configurable capabilities. Such devices incur enormous costs if they interface with clouds directly due to their limited internal capabilities. As a result, performing such resource-intensive processes to communicate with clouds requires certain intermediary deployments and complicated protocols [79].

7 Conclusion

At-risk patients can be managed outside of traditional clinical settings through IoT- edge health systems. Since a service flaw could have fatal repercussions for the patients, these systems are required to provide a high quality of experience. Artificial intelligence, such as deep learning, can perform satisfactorily in terms of decision-making since a large amount of data can be provided to the classification algorithm. These techniques can be fully implemented in the cloud machines in the traditional cloud-based IoT systems.

Due to the service's reliance on the quality of the internet connection quality, these solutions are inadequate for a time-sensitive health application. Other options include edge-based systems that are fully distributed. By utilizing healthcare information and data processing on the edges of networks, healthcare framework-based service provisioning can enable smart healthcare applications and systems.

The demand for long-term smart healthcare solutions is driven by the requirement for a patient care system that is cost-effective.

References

1. Adly, A.S., Adly, M.S., Adly, A.S.: Telemanagement of home-isolated COVID-19 patients using oxygen therapy with noninvasive positive pressure ventilation and physical therapy techniques: randomized clinical trial. J. Med. Internet Res. **23**(4), e23446 (2021)
2. Lehmann, U., Dieleman, M., Martineau, T.: Staffing remote rural areas in middle-and low-income countries: a literature review of attraction and retention. BMC Health Serv. Res. **8**, 1–10 (2008)
3. Pathinarupothi, R.K., Durga, P., Rangan, E.S.: IoT-based smart edge for global health: remote monitoring with severity detection and alerts transmission. IEEE Internet Things J. **6**(2), 2449–2462 (2018)
4. Zhou, Z., et al.: Edge intelligence: paving the last mile of artificial intelligence with edge computing. Proc. IEEE. **107**(8), 1738–1762 (2019)
5. Ray, P.P., Dash, D., De, D.: Intelligent internet of things enabled edge system for smart healthcare. Natl. Acad. Sci. Lett. **44**, 325–330 (2021)
6. Adly, A.S., Adly, A.S., Adly, M.S.: Healthcare technology trade-offs for IoT ecosystems from a developing country perspective: case of Egypt. In: Healthcare Paradigms in the Internet of Things Ecosystem, pp. 313–331. Elsevier (2021)
7. Kamruzzaman, M.: E-crime management system for future smart city. In: Data Processing Techniques and Applications for Cyber-Physical Systems (DPTA 2019). Springer (2020)
8. Salagare, S., Prasad, R.: An overview of internet of dental things: new frontier in advanced dentistry. Wirel. Pers. Commun. **110**, 1345–1371 (2020)
9. Adly, M.S., Adly, A.S., Adly, A.S., Carayon, D., Estephan, E., Cuisinier, F.: Innovative device for in-vivo quantitative assessment of different wire materials using 3D models. J. Sci.: Adv. Mater. Devices, **9**(1), 100667 (2024)
10. Ayoub, A., Pulijala, Y.: The application of virtual reality and augmented reality in Oral & Maxillofacial Surgery. BMC Oral Health. **19**, 1–8 (2019)
11. Shi, W., Dustdar, S.: The promise of edge computing. Computer. **49**(5), 78–81 (2016)
12. Awad, A.I., et al.: Utilization of mobile edge computing on the internet of medical things: a survey. ICT Express. **9**, 473–485 (2022)
13. Liu, G., Shen, H., Wang, H.: Towards long-view computing load balancing in cluster storage systems. IEEE Trans Parallel Distrib Syst. **28**(6), 1770–1784 (2016)
14. Adly, A.S., et al.: Effectiveness of laser acupuncture as adjunctive therapy for limited range of motion in rheumatoid arthritis. Laser Phys. **29**(2), 025601 (2018)
15. Hao, Y., et al.: Smart-Edge-CoCaCo: AI-enabled smart edge with joint computation, caching, and communication in heterogeneous IoT. IEEE Netw. **33**(2), 58–64 (2019)

16. Kamruzzaman, M., Alrashdi, I., Alqazzaz, A.: New opportunities, challenges, and applications of edge-AI for connected healthcare in internet of medical things for smart cities. J. Healthcare Eng. **2022**, 2950699 (2022)
17. Alrazgan, M.: Internet of medical things and edge computing for improving healthcare in smart cities. Math. Probl. Eng. **2022**, 1–10 (2022)
18. Wang, H., et al.: Healthedge: task scheduling for edge computing with health emergency and human behavior consideration in smart homes. In: 2017 IEEE International Conference on Big Data (Big Data). IEEE (2017)
19. Debauche, O., et al.: Edge AI-IoT pivot irrigation, plant diseases, and pests identification. Procedia Comput. Sci. **177**, 40–48 (2020)
20. Rahman, M.A., Hossain, M.S.: An internet-of-medical-things-enabled edge computing framework for tackling COVID-19. IEEE Internet Things J. **8**(21), 15847–15854 (2021)
21. Singh, R., et al.: Identification of the key parameters for computational offloading in multi-access edge computing. In: 2020 IEEE Cloud Summit. IEEE (2020)
22. Gill, S.S., Garraghan, P., Buyya, R.: ROUTER: fog enabled cloud based intelligent resource management approach for smart home IoT devices. J. Syst. Softw. **154**, 125–138 (2019)
23. Nabavi, S., et al.: Seagull optimization algorithm based multi-objective VM placement in edge-cloud data centers. Internet Things Cyber-Phys. Syst. **3**, 28–36 (2023)
24. Iftikhar, S., et al.: FogDLearner: a deep learning-based cardiac health diagnosis framework using fog computing. In: Proceedings of the 2022 Australasian Computer Science Week, pp. 136–144 (2022)
25. Chen, X., et al.: Efficient multi-user computation offloading for mobile-edge cloud computing. IEEE/ACM Trans. Networking. **24**(5), 2795–2808 (2015)
26. Teoh, Y.K., Gill, S.S., Parlikad, A.K.: IoT and fog computing based predictive maintenance model for effective asset management in industry 4.0 using machine learning. IEEE Internet Things J. (2021). https://doi.org/10.1109/JIOT.2021.3050441
27. Shi, Y., et al.: Communication-efficient edge AI: algorithms and systems. IEEE Commun Surv Tutor. **22**(4), 2167–2191 (2020)
28. Adly, A.S.: Climate change and energy decision aid systems for the case of Egypt. In: Climate Change and Energy Dynamics in the Middle East: Modeling and Simulation-Based Solutions, pp. 79–107 (2019)
29. Jia, M., Cao, J., Yang, L.: Heuristic offloading of concurrent tasks for computation-intensive applications in mobile cloud computing. In: 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE (2014)
30. Stankovic, J.A.: Research directions for cyber physical systems in wireless and mobile healthcare. ACM Trans. Cyber-Phys. Syst. **1**(1), 1–12 (2016)
[MathSciNet]

31. Kalapothas, S., Flamis, G., Kitsos, P.: Efficient edge-AI application deployment for FPGAs. Information. **13**(6), 279 (2022)
32. Sriraghavendra, M., et al.: DoSP: a deadline-aware dynamic service placement algorithm for workflow-oriented IoT applications in fog-cloud computing environments. In: Energy Conservation Solutions for Fog-Edge Computing Paradigms, pp. 21–47 (2022)
33. Letaief, K.B., et al.: Edge artificial intelligence for 6G: vision, enabling technologies, and applications. IEEE J Sel Areas Commun. **40**(1), 5–36 (2021)
34. Nabavi, S.S., et al.: TRACTOR: traffic-aware and power-efficient virtual machine placement in edge-cloud data centers using artificial bee colony optimization. Int. J. Commun. Syst. **35**(1), e4747 (2022)
35. Malik, S.U., et al.: EFFORT: energy efficient framework for offload communication in mobile cloud computing. Softw. Pract. Experience. **51**(9), 1896–1909 (2021)
36. Kaur, A., Singh, V., Gill, S.S.: The future of cloud computing: opportunities, challenges and research trends. In: 2018 2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC) I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), 2018 2nd International Conference on. IEEE (2018)
37. Han, B., et al.: Context-awareness enhances 5G multi-access edge computing reliability. IEEE Access. **7**, 21290–21299 (2019)
38. Ferdowsi, A., Challita, U., Saad, W.: Deep learning for reliable mobile edge analytics in intelligent transportation systems: an overview. IEEE Veh. Technol. Mag. **14**(1), 62–70 (2019)
39. Rapuzzi, R., Repetto, M.: Building situational awareness for network threats in fog/edge computing: emerging paradigms beyond the security perimeter model. Futur. Gener. Comput. Syst. **85**, 235–249 (2018)
40. Xing, K., et al.: Design of a wearable rehabilitation robotic hand actuated by pneumatic artificial muscles. In: 2009 7th Asian Control Conference. IEEE (2009)
41. Adly, A.S., Adly, A.S., Adly, M.S.: Effects of laser acupuncture tele-therapy for rheumatoid arthritis elderly patients. Lasers Med. Sci. **37**(1), 499–504 (2022)
42. Sutherland, A.E., Stickland, J., Wee, B.: Can video consultations replace face-to-face interviews? Palliative medicine and the Covid-19 pandemic: rapid review. BMJ Support. Palliat. Care. **10**(3), 271–275 (2020)
43. Adly, A.S., Haggag, M.H., Mostafa, M.-S.M.: Low intensity laser irradiation influence proliferation of mesenchymal stem cells: comparison of experimental data to intelligent agent-based model predictions. In: Applied Methods and Techniques for Mechatronic Systems: Modelling, Identification and Control, pp. 293–306. Springer (2013)
44. Adly, A.S., Haggag, M.H., Mostafa, M.-S.M.: Laser promotes proliferation of stem cells: a comprehensive case study consolidated by intelligent agent–based model predictions. In: Case Studies in Intelligent Computing: Achievements and Trends, p. 31 (2014)

45. Adly, M.S., Adly, A.S., Adly, A.S.: Assessment of early orthodontic treatment on functional shifts by telemonitoring mandibular movements using a smart phone. J. Telemed. Telecare. **26**(3), 150–160 (2020)

46. Ahmad, I., et al.: The challenges of artificial intelligence in wireless networks for the internet of things: exploring opportunities for growth. IEEE Ind. Electron. Mag. **15**(1), 16–29 (2020)

47. Keller, J.M., Liu, D., Fogel, D.B.: Fundamentals of Computational Intelligence: Neural Networks, Fuzzy Systems, and Evolutionary Computation. Wiley (2016)

48. Adly, A.S.: Integrating vehicular technologies within the IoT environment: a case of Egypt. In: Connected Vehicles in the Internet of Things: Concepts, Technologies and Frameworks for the IoV, pp. 85–100 (2020)

49. Adly, A.S.: Technology trade-offs for IIoT systems and applications from a developing country perspective: case of Egypt. In: The Internet of Things in the Industrial Sector: Security and Device Connectivity, Smart Environments, and Industry 4.0, pp. 299–319 (2019)

50. Chaqfeh, M.A., Mohamed, N.: Challenges in middleware solutions for the internet of things. In: 2012 International Conference on Collaboration Technologies and Systems (CTS). IEEE (2012)

51. Vermesan, O., et al.: Internet of things strategic research and innovation agenda. In: Internet of Things: Converging Technologies for Smart Environments and Integrated Ecosystems, pp. 7–152 (2013)

52. Adly, A.S., Adly, M.S., Cuisinier, F., Egea, J.C., Panayotov, I., Adly, A.S., Malthiery, E.: Laser-Induced Blood Coagulation for Surgical Application: A Scoping Review. In: Seminars in Thrombosis and Hemostasis. Thieme Medical Publishers, Inc. (2023, August)

53. Vaquero, L.M., Rodero-Merino, L.: Finding your way in the fog: towards a comprehensive definition of fog computing. ACM SIGCOMM Comput. Commun. Rev. **44**(5), 27–32 (2014)

54. Alavi, M., Leidner, D.E.: Knowledge management and knowledge management systems: conceptual foundations and research issues. In: MIS Quarterly, pp. 107–136 (2001)

55. Compton, M., et al.: The SSN ontology of the W3C semantic sensor network incubator group. J. Web Semant. **17**, 25–32 (2012)

56. Barnaghi, P., et al.: Semantics for the Internet of Things: early progress and back to the future. Int. J. Semant. Web Inf. Syst. (IJSWIS). **8**(1), 1–21 (2012)

57. Singh, A., Chatterjee, K.: Securing smart healthcare system with edge computing. Comput. Secur. **108**, 102353 (2021)

58. Singh, U., et al.: Review of software defined networking: applications, challenges and advantages. In: Inventive Computation Technologies 4. Springer (2020)

59. Adly, A.S., Adly, A.S., Adly, M.S.: Approaches based on artificial intelligence and the internet of intelligent things to prevent the spread of COVID-19: scoping review. J. Med. Internet Res. **22**(8), e19104 (2020)

60. Adly, A.S., et al.: A novel approach utilizing laser acupuncture teletherapy for management of elderly-onset rheumatoid arthritis: a randomized clinical trial. J. Telemed. Telecare. **27**(5), 298–306 (2021)
61. Gao, Y., Al-Sarawi, S.F., Abbott, D.: Physical unclonable functions. Nat. Electron. **3**(2), 81–91 (2020)
62. Abdellatif, A.A., et al.: Edge computing for smart health: context-aware approaches, opportunities, and challenges. IEEE Netw. **33**(3), 196–203 (2019)
63. Dilawar, N., et al.: Blockchain: securing internet of medical things (IoMT). Int. J. Adv. Comput. Sci. Appl. **10**(1) (2019). https://doi.org/10.14569/IJACSA.2019.0100110
64. Adly, M.S., et al.: Can paracetamol lower stress and anxiety by blunting emotions during and after computer guided dental implant surgeries? Findings from a randomized crossover clinical trial. J. Dent. Sci. **16**(1), 137–144 (2021)
65. Adly, M.S., et al.: Can combining low level laser therapy with computer guided flapless Piezosurgical osteotomy achieve a painless implant surgery? Findings of Split mouth randomized controlled trial. J. Evid. Based Dent. Pract. **22**(3), 101730 (2022)
66. Raja'a, M., Farid, F.: Computer-based technologies in dentistry: types and applications. J. Dent. (Tehran, Iran). **13**(3), 215 (2016)
67. Adly, M.S., et al.: Prevention and repair of orthodontically induced root resorption using ultrasound: a scoping review. In: Expert Review of Medical Devices (2023) (just-accepted)
68. Giraudeau, N., et al.: The year of teledentistry in France? J. Int. Soc. Telemed. eHealth. **7**, e8 (1–5) (2019)
69. Adly, A.S., et al.: Computational and theoretical concepts for regulating stem cells using viral and physical methods. In: Machine Learning and Systems Engineering, pp. 533–546 (2010)
70. Adly, A.S., Adly, M.S., Adly, A.S.: Authors reply to: research using social media to recruit research participants should proceed with caution. Comment on "Telemanagement of home-isolated COVID-19 patients using oxygen therapy with noninvasive positive pressure ventilation and physical therapy techniques: randomized clinical trial". J. Med. Internet Res. **24**(5), e37413–e37413 (2022)
71. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature. **521**(7553), 436–444 (2015)
72. Caprolu, M., et al.: Edge computing perspectives: architectures, technologies, and open security issues. In: 2019 IEEE International Conference on Edge Computing (EDGE). IEEE (2019)
73. Adly, A.S., Aboutabl, A.E., Ibrahim, M.S.: Modeling of gene therapy for regenerative cells using intelligent agents. In: Software Tools and Algorithms for Biological Systems, pp. 317–325 (2011)
74. Dong, P., et al.: Edge computing based healthcare systems: enabling decentralized health monitoring in internet of medical things. IEEE Netw. **34**(5), 254–261 (2020)

75. Kakhi, K., et al.: The internet of medical things and artificial intelligence: trends, challenges, and opportunities. Biocybern. Biomed. Eng. **42**(3), 749–771 (2022)
76. Cui, J., et al.: An efficient message-authentication scheme based on edge computing for vehicular ad hoc networks. IEEE Trans. Intell. Transp. Syst. **20**(5), 1621–1632 (2018)
77. Hakak, S., et al.: A framework for edge-assisted healthcare data analytics using federated learning. In: 2020 IEEE International Conference on Big Data (Big Data). IEEE (2020)
78. Adly, A.S., et al.: Laser acupuncture versus reflexology therapy in elderly with rheumatoid arthritis. Lasers Med. Sci. **32**, 1097–1103 (2017)
79. Chen, M., et al.: Edge cognitive computing based smart healthcare system. Futur. Gener. Comput. Syst. **86**, 403–411 (2018)

OceanofPDF.com

Index

A

AI accelerators
  GPU 9, 38, 43, 45–47, 50, 52, 53, 119142, 143, 353
  TPU 38, 43, 45–47, 50, 52, 53, 84
  VPU 38, 43, 45–47, 50, 52, 53
AI inference 37–64, 71, 73, 255
Air quality prediction/monitoring 303, 335, 336, 339, 341–347, 350, 352, 354–357359, 360
Analytics v, 5, 15, 17, 40, 61, 69, 7582, 83, 114, 115, 117, 119, 122, 126140, 145, 150, 155, 202, 206, 207, 210214, 226, 238–240, 242, 244, 245, 247, 248250–255, 261, 296, 303, 304, 307, 308, 315–329352, 360, 379
Anomaly detection v, 93, 140, 173–194, 252, 253, 255, 256259, 261, 262
Artificial intelligence (AI) 3, 37, 67, 112, 150, 208, 238, 292315, 336, 366
Autonomous navigation system v, 315–329

C

Cloud 3, 40, 68, 111, 149, 190, 201, 236266, 293, 323, 335, 366
Concept drift 174–176, 179, 182, 185–188, 190–192, 194
Cyber-physical system (CPS) 117

D

Data as a Product 216–220
Data exchange v, 8, 89, 173, 201–229, 234, 240, 247258, 292, 294, 296, 308, 310, 324
Data Fabric 202, 215–218, 221–223, 225, 228
Data Spaces 201, 202, 208–212, 214, 215, 223, 224, 227228
Deep learning (DL) 13, 20, 21, 23, 29, 37, 40, 4546, 68, 78, 79, 95, 96, 105, 118119, 131, 143, 173, 174, 187–190, 193, 194252, 259, 325, 353, 355, 359, 379, 381

E

Edge, 3, 37, 68, 111, 149, 190, 202, 233267, 293, 315, 336, 367
Edge AI 37–64, 72, 83–86, 123–125, 139, 255–257, 335–361, 371–372
Edge-cloud continuum 63, 201–229, 253
Edge computing, 3, 38, 68, 112, 149, 173, 201, 234267, 292, 315, 336, 366

Edge intelligence v, vi, 3–32, 39–42, 77–81, 83–85, 245, 254256, 259, 261, 265–286, 316, 317, 323–329
Edge intelligence systems v, 3–32
Energy consumption 75, 82, 105, 107, 128, 130, 133, 135227, 239, 251, 253, 256–258, 260, 267, 273–277279–283, 285, 286, 292, 294, 345–347, 351, 378
Evaluation 3–32, 43, 94, 125, 126, 131, 168, 174214, 223, 265–267, 269–274, 277, 282, 283, 285286, 324, 340

I

Industry 5.0 111–145
Integration 4, 21, 30, 37, 39, 51, 64, 7677, 79, 83, 84, 91, 101, 102, 104–107112, 114–117, 119, 120, 122, 123, 126, 133139, 144, 145, 166, 201–204, 206, 207, 209210, 214, 216, 217, 221, 223, 225, 226228, 233, 236, 238, 239, 245, 248–254, 257–261291, 293, 309–311, 316, 324, 326–328, 352, 353358, 367, 373
Intelligence 3, 40, 67, 112, 150, 224, 255, 292316, 350, 366
Intelligent transportation system v, 207, 254, 291–311
Internet of Things (IoT) 27, 40, 68, 111, 152, 173, 201, 233265, 292, 315, 335, 366

M

Machine learning (ML) 3, 37, 67, 112, 173, 206, 248, 265292, 325, 335, 372
Modeling v, 3–32, 118, 120, 221, 237, 355, 379

N

Network bandwidth 16, 71, 95, 267, 274, 275, 282, 283286, 380
Neural network v, 13, 20, 45, 68, 188, 265–286, 372
Next generation computing system 67–107

O

Open Digital Rights Language (ODRL) 150–153, 157–168, 170

P

Policy 8, 77, 118, 150, 208, 241, 304, 320
Power consumption 8–10, 18, 44, 84, 85, 248, 267, 268273–284, 286, 343, 347, 367, 374, 378
Privacy, 13, 40, 68, 113, 152, 203, 233, 266299, 319, 352, 367

Privacy-preserving machine learning 21, 84

R
Remote dental services 365–381
Remote medical services 365–381

S
Security, 14, 40, 68, 113, 149, 203, 233, 266292, 320, 352, 372
Sensor, 3, 68, 111, 154, 173, 203, 235, 265292, 315, 335, 366
Service, 3, 38, 67, 114, 150, 202, 233, 265295, 323, 336, 365
Simulation 3–32, 49, 67, 117, 118, 122, 127–133, 143237–238, 319
Smart applications 81–82
Smart environment 323
Smart parking 301, 307–310
Smart roads 301, 305–308, 310, 311
Smart streetlights 303

T
Telemedicine 139, 375
Trust v, 85, 88–91, 93, 94, 96–97, 104, 105136–138, 157, 201–229, 233–262, 285, 299, 376

U
Unsupervised learning 180

W
Waste incineration pollution 335–361

OceanofPDF.com